NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

CAN SUBJECTS BE GUIDED TO OPTIMAL DECISIONS? THE USE OF A REAL-TIME TRAINING INTERVENTION MODEL

by

Travis D. Carlson

June 2016

Thesis Advisor:	Quinn Kennedy
Co-Advisor:	Lee Sciarini

This thesis was performed at the MOVES Institute

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704–0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** June 2016	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE CAN SUBJECTS BE GUIDED TO OPTIMAL DECISIONS? THE USE OF A REAL-TIME TRAINING INTERVENTION MODEL	**5. FUNDING NUMBERS**
6. AUTHOR(S) Travis D. Carlson	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number NPS.2016.0017-IR-EP4-A

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

Effective decision-making is a hallmark of military leadership, and development of decision makers is critical to military strategy. The Cognitive Alignment with Performance-Targeted Training Intervention Model (CAPTTIM) was developed to aid training of optimal decision-making. Cognitive state suggests a subject is exploring the decision environment as opposed to exploiting it, and decision performance classifies whether a subject is making optimal decisions. Using a color-coded structure combining cognitive state and decision performance, CAPTTIM indicates whether those factors are aligned for optimal decision-making—exploiting the environment and making optimal decisions—or not. The focus of this thesis was to identify each subject's CAPTTIM status in real time and, when decision performance was misaligned, provide feedback to influence the subject's future decisions.

Through a human-subject experiment (n = 34), we classified decision-makers' CAPTTIM status in real time. We randomly assigned 17 subjects to receive tailored feedback during execution of a decision task (feedback group), and trend analysis reveals the feedback group to be more likely to reach optimal decisions than a control group.

These results imply that training systems could be tailored to the individual and that methods used to instruct effective decision-making may expand to include real-time understanding and intervention.

14. SUBJECT TERMS decision making, optimal decision making, training, real-time data capture, intervention	**15. NUMBER OF PAGES** 119
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

THIS PAGE INTENTIONALLY LEFT BLANK

CAN SUBJECTS BE GUIDED TO OPTIMAL DECISIONS? THE USE OF A REAL-TIME TRAINING INTERVENTION MODEL

Travis D Carlson
Major, United States Marine Corps
B.S., University of Oregon, 1995

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN
MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION (MOVES)**

from the

**NAVAL POSTGRADUATE SCHOOL
June 2016**

Approved by: Dr. Quinn Kennedy
Thesis Advisor

LT Lee Sciarini
Co-Advisor

Dr. Peter Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Effective decision-making is a hallmark of military leadership, and development of decision makers is critical to military strategy. The Cognitive Alignment with Performance-Targeted Training Intervention Model (CAPTTIM) was developed to aid training of optimal decision-making. Cognitive state suggests a subject is exploring the decision environment as opposed to exploiting it, and decision performance classifies whether a subject is making optimal decisions. Using a color-coded structure combining cognitive state and decision performance, CAPTTIM indicates whether those factors are aligned for optimal decision-making—exploiting the environment and making optimal decisions—or not. The focus of this thesis was to identify each subject's CAPTTIM status in real time and, when decision performance was misaligned, provide feedback to influence the subject's future decisions.

Through a human-subject experiment (*n* = 34), we classified decision-makers' CAPTTIM status in real time. We randomly assigned 17 subjects to receive tailored feedback during execution of a decision task (feedback group), and trend analysis reveals the feedback group to be more likely to reach optimal decisions than a control group.

These results imply that training systems could be tailored to the individual and that methods used to instruct effective decision-making may expand to include real-time understanding and intervention.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

Above all, I would like to thank my wife, Karena, and my children, Zoë and Anders, for seeing me through the completion of my master's degree. Without their patience and support, I would not have been able to devote the time needed to complete my studies and research.

I also would like to thank Dr. Quinn Kennedy for her unwavering support and guidance. LT Lee Sciarini also added valuable insight to the process. Without their acumen and knowledge, this thesis would not have been possible.

In addition, I would like to acknowledge MAJ Cardy Moten. He provided critical assistance writing the Convoy Task Python code that performed the real-time data capture in support of this thesis. Without his innovation and willingness to instruct this study, the project would have stalled before it started.

Finally, I would like to thank the Naval Postgraduate School professors who are dedicated to students and quality education. Their tailored guidance and assistance helped make this thesis possible.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Military leaders will affirm that, of the myriad critical tasks required of military personnel, decision-making is a crucial skill. For example, decisions made by junior officers and enlisted service members often have life-or-death consequences, and the outcomes of those decisions can have strategic implications capable of impacting military and government courses of action well beyond a particular moment of action. Thus, the need to understand how effective decisions are made is critical to the continued success of our armed forces. Military leadership recognizes the importance of agile, adaptive thinkers. The U.S. Army and U.S. Marine Corps have each issued strategic guidance initiatives directing efforts to improve decision-making. The Army's Human Dimension Strategy 2015 directs the Service to "improve the decision-making ability and ethical conduct of Soldiers and Army Civilians through individual and collective learning programs that challenge Army Professionals in complex operational and ethical situations" (Odierno & McHugh, 2015, p. 7). Similarly, Marine Corps Science and Technology Objective (Training and Education) -1 states that the Corps aims to "develop capabilities to enhance cognitive, relational, and perceptual skills for small unit leaders to make effective decisions in complex environments; enhancements include attention control, expertise, metacognitive skills, and accelerated learning outcomes" (U.S. Marine Corps, 2012, p. 34). However, as military experience is hard won—specifically combat experience where a leader may ever have only one chance to learn from a decision—understanding decision-making in a training and educational environment has become the focus of increased study (Bechara, Damasio, Tranel & Damasio, 1997; Critz, 2015; Kennedy, Nesbitt, Alt & Fricker, 2015; Nesbitt, Kennedy, Alt, Yang, Fricker, Appleget, Huston, Patton & Whitaker, 2013). This thesis is one small part of larger efforts striving to understand the decision-making processes, and improve decision-making among service members to increase the combat effectiveness of the military.

Combat always has been complex; however, that complexity increases significantly when service members are confronted with challenges beyond basic weapons employment, tactics, and lower-level strategy. History is rife with leaders using measures of performance, such as enemy attrition, to draw conclusions about the effectiveness of their operations; and discovering too late that the information being used to drive decisions was not pertinent to the long-term outcome of the conflict. Modern warfighters are routinely confronted with complex battlefield situations involving noncombatants, irregular threats, humanitarian crises and even governance. While not every decision can be perfect, and military leaders will rarely have perfect information on which to base their decisions, it is important that warfighters possess the cognitive flexibility to recognize a changing situation and use the experience gained to adjust the decision-making process. If we, as military leaders, better understand decision performance and an optimal decision making process, we can train the next generation of leaders to make the best possible decision their environment presents.

1. Cognitive Abilities Needed to Achieve Optimal Military Decision-Making

Reinforcement learning—the ability to learn from trial and error—is a cognitive characteristic necessary for individuals to achieve optimal decision-making (Sutton & Barto, 1998). Decisions in the military environment often involve a degree of uncertainty. When intelligence estimates of an enemy location or the strength of an enemy force are not well established, a military professional is still faced with a decision of how (or whether) to act against the enemy, for action is surely still required. Thus, the action relies upon the decision maker’s accumulated experience and the reinforcement learning that has been accrued through the experience, whether those decisions and learning were optimal or not One existing evaluation of reinforcement learning is the Iowa

Gambling Task (IGT) (Bechara, Damasio, Damasio & Anderson, 1994). The IGT has been widely applied and documented in numerous psychology studies (Krain, Wilson, Arbuckle, Castellanos & Milham, 2006) and will be discussed further here as it serves as the basis of a military-themed reinforcement learning, cognitive analysis tool.

A second characteristic of optimal decision-making is cognitive flexibility. As we expect our military decision makers to learn from experience, we assume that the learning is incorporated into future decision-making and that existing problem solving strategies are adapted based upon the information being provided. That is, when a situation, or information within the problem space, changes “an individual needs to realize that the situation has changed in order to be able to ‘log out’ of the automatic processing mode and come into the controlled processing mode” (Canas, Quesada, Antoli & Fajardo, 2003, p. 484). This ability to enter the controlled processing mode is cognitive flexibility. In this thesis, we hope to influence the decision makers while they complete a military version of the IGT to bring them into this controlled processing mode, and then determine whether this cognitive flexibility can be leveraged toward optimal decisions.

2. Current Military Decision-Making Instruction

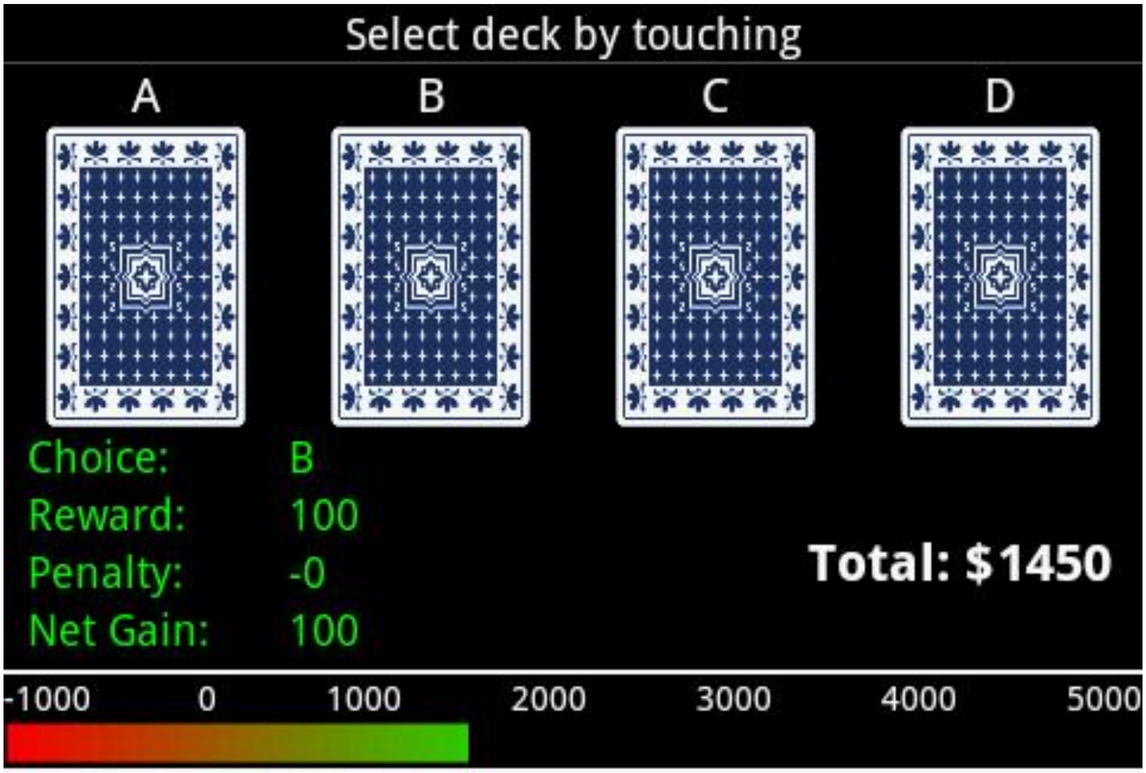
The current operational environment offers increased opportunity to understand decision-making and develop programs to more effectively train this critical skill. After many years of combat operations, long deployments in complex environments, and dynamic, difficult decision-making, the military has a unique opportunity to use the experience gained to understand how this population of experienced decision makers functions; toward understanding factors such as their cognitive state during the decision making process. For example, when do experienced decision makers feel that they need to learn more about the environment and when do they feel that they know the environment well enough to make optimal decisions? This opportunity may allow less-experienced personnel, and their instructors to understand cognitive state and thus leverage situationally dependent information to make optimal decisions. Furthermore, those agencies tasked with educating on and instructing for decision-making can tailor instruction to the individual decision-maker. The Basic School (TBS) is the U.S. Marine Corps' entry-level training and education venue for newly commissioned officers. Every Marine officer—whether future armor officer, aviator, infantry officer, lawyer or logistician—spends six months at this school being educated and evaluated on tactics and leadership, of which decision-making is a key facet. TBS is just one example of an institution that applies significant effort to ensure junior officers have an appreciation for *how* to make effective decisions. As a former instructor at TBS, the author can confirm that the current method of evaluating the effectiveness of the student's decisions relies upon subject matter expertise and direct evaluation of the trainee. Direct observation, with little appreciation for the trainee's cognitive state or decision-making history leaves much to chance when trying to train and educate the military's future key decision makers. As we define it, the cognitive state of a subject, or trainee, will indicate whether he or she is exploring or exploiting the decision environment; that is, whether the decision maker believes they have all the information required to make optimal decisions. Thus, understanding the trainee's cognitive state may help to produce exercises that will effectively instruct on the art and science of decision-making. The focus of this thesis was to explore whether a trainee's cognitive state and decisions can be effectively influenced, in real time, toward the optimal set of decisions.

B. DECISION MAKING

As stated in previous work, "current reinforcement-learning tests, which are typically computerized laboratory tests, do not account for the stress, uncertainty, and high-risk conditions of decisions made in combat" (Nesbitt et. al, 2013, p. 3). We will explore an established psychological decision-making test, its modification to a more military-relevant decision task, and the categorization of decision-maker cognitive state and decision performance scores into a single color-coded categorization in the Cognitive Alignment with Performance Targeted Training Intervention tool.

1. Iowa Gambling Task

The IGT is an established psychological test in which subjects make a series of decisions and the effect of reinforcement learning can be studied based upon the patterns of decisions observed (Bechara et al., 1994). Subjects are presented with a computer screen on which four decks of cards are displayed face down, and are told to choose cards to optimize their long-term gain. (See Figure 1.)



Select deck by touching

Choice: B
Reward: 100
Penalty: -0
Net Gain: 100

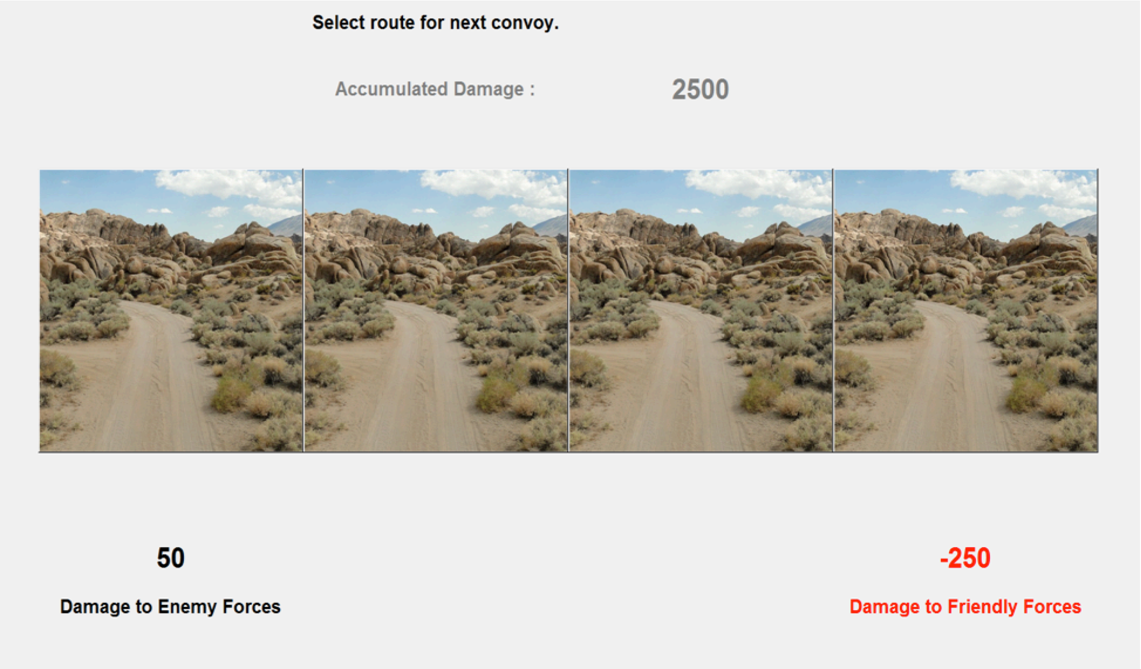
Total: $1450

Figure 1. The Iowa Gambling Task Screenshot. Source: Sacchi (2015)

The subject begins the trial with a loan of an imaginary $2000. Each card selected results in some amount of gain and some amount of loss such that, over time, the subjects can conjecture the net gain or net loss after multiple selections and careful observation of gain/loss patterns. As success is defined as ending a set number of trials (usually 100 – 200 individual selections) with the most money possible, “participants can succeed on the IGT only when they learn to forgo high immediate rewards and prefer the safe options over the risky options” (Steingroever and Wetzels, Horstmann, Neumann & Wagenmakers, 2013, p. 180). What is initially unknown to the subject is that the payouts are predetermined, and further, certain decks will always provide a higher long-term payout than others. Ultimately, the subject is meant to recognize that decks A and B are long-term losers; although in the first few selections these decks reward the subject, decks A and B are heavily penalized later resulting in a net loss over 10 or 15 selections. Previous studies have concluded, “subjects must rely on their ability to develop an estimate of which decks are risky and which are profitable in the long run” (Bechara et al. 1994, p. 13). Eventually, a subject should realize that despite smaller payouts-per-trial from decks C and D the long-term payout is greater.

2. Convoy Task

We will build on the foundation of the IGT; past work at the Naval Postgraduate School (NPS) has converted the same decision-making evaluation approach to a military-relevant decision making tool called the Convoy Task. (See Figure 2).



Select route for next convoy.

Accumulated Damage : 2500

50

Damage to Enemy Forces

-250

Damage to Friendly Forces

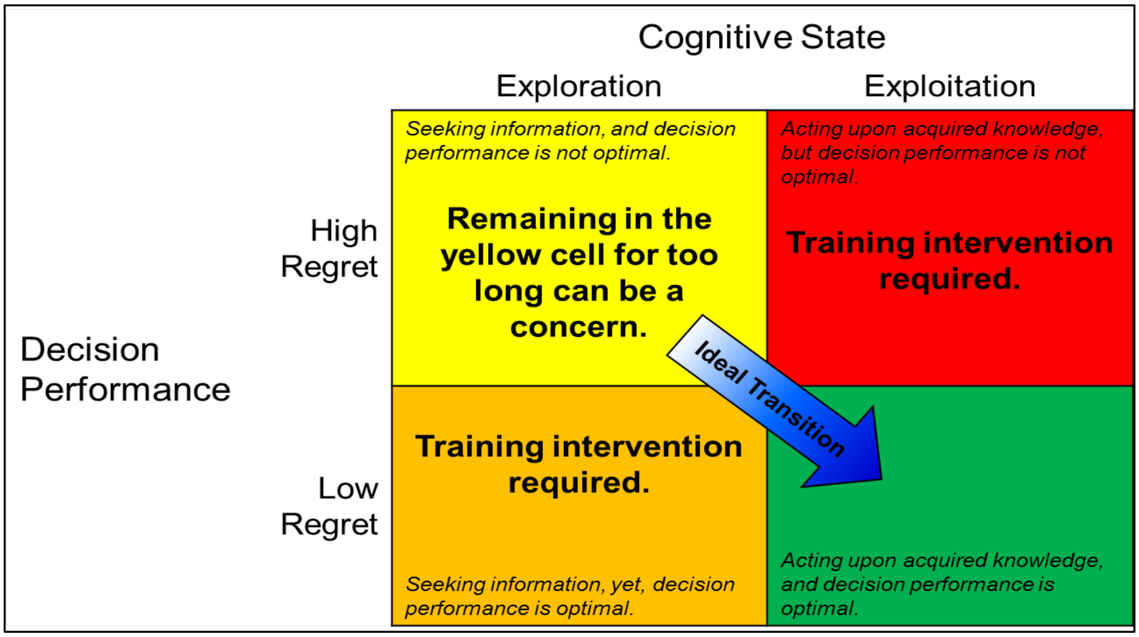
The decision just executed by this subject has resulted in a gain of 50 damage points (Damage to Enemy Forces) and a loss of 250 damage points (Damage to Friendly Forces) for a net change to Accumulated Damage of -200 points.

Figure 2. Convoy Task Screen.

The creators of the Convoy Task state that “this new task focuses on high stakes and uncertain environments particular to military decision making condition and retains essential characteristics of the foundational task and gives insight into reinforcement learning of military decision makers” (Nesbitt et. al, 2013, p. 10). As opposed to a monetary reward and penalty system, the creators used a more military-relevant scoring system; damage to enemy and friendly forces. Damage to Enemy Forces is the reward and adds to the running score, termed Accumulated Damage, which stands in for the $2,000 loan amount in the IGT. The penalty is termed Damage to Friendly Forces, and it subtracts from Accumulated Damage (Nesbitt et al., 2013). And rather than identical decks of cards, subjects are presented with four identical photos of a non-descript road that might depict a convoy route. Past data collected from 34 subjects confirmed that the Convoy Task requires reinforcement learning to effectively add to the total Accumulated Damage score (Kennedy et al., 2015).

3. Cognitive Alignment with Performance Targeted Training Intervention

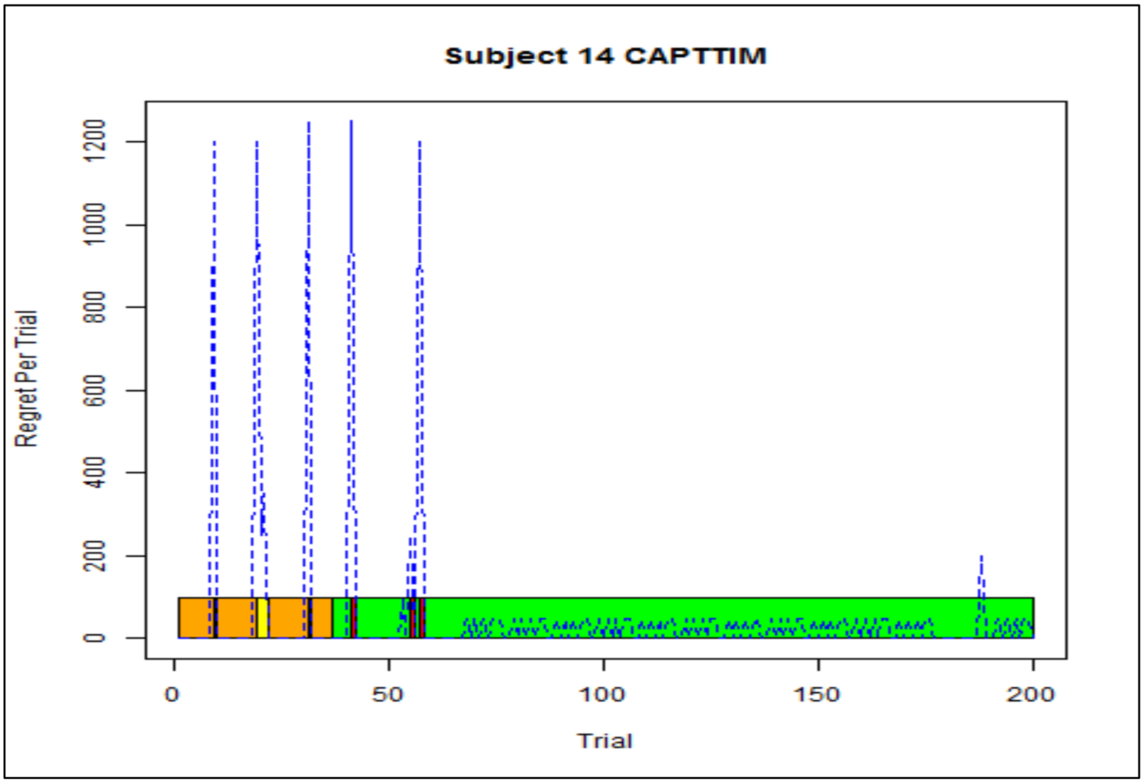
Efforts at NPS by Kennedy et al. (2015) resulted in a model called Cognitive Alignment with Performance Targeted Training Intervention Model (CAPTTIM) that places subjects into one of four color-coded categories based upon cognitive state and decision performance. This model distinguishes between two subject cognitive states: exploration (feeling that one has not figured out the task and needs to explore the environment more) and exploitation (where a subject thinks that they have mastered the task and is acting upon acquired knowledge). The model then determines whether cognitive state is aligned or misaligned with observed decision performance. (See Figure 3). CAPTTIM utilizes simple behavioral measures to characterize cognitive state and decision performance. It uses variability in latency from decision to decision to determine whether the trainee's cognitive state is exploration (large latency variability) or exploitation (small latency variability). Decision performance is measured by regret, the difference between the trainee's decision and the optimal decision, given perfect knowledge of the task. High regret indicates poor decision performance; low regret indicates near optimal decision performance. Thus, accumulated regret provides a measure of how far off the trainee is from the optimal decision path. In the NPS master's thesis from 2015, Critz established the threshold delineating between high and low regret of each decision during the same decision-making task and concluded, "by looking at a common reinforcement learning task, modified for the military domain the thesis team was able to investigate and better understand a subject's decision-making pattern" (Critz, 2015, p. 50).



	Exploration	Exploitation
High Regret	*Seeking information, and decision performance is not optimal.* **Remaining in the yellow cell for too long can be a concern.**	*Acting upon acquired knowledge, but decision performance is not optimal.* **Training intervention required.**
Low Regret	**Training intervention required.** *Seeking information, yet, decision performance is optimal.*	*Acting upon acquired knowledge, and decision performance is optimal.*

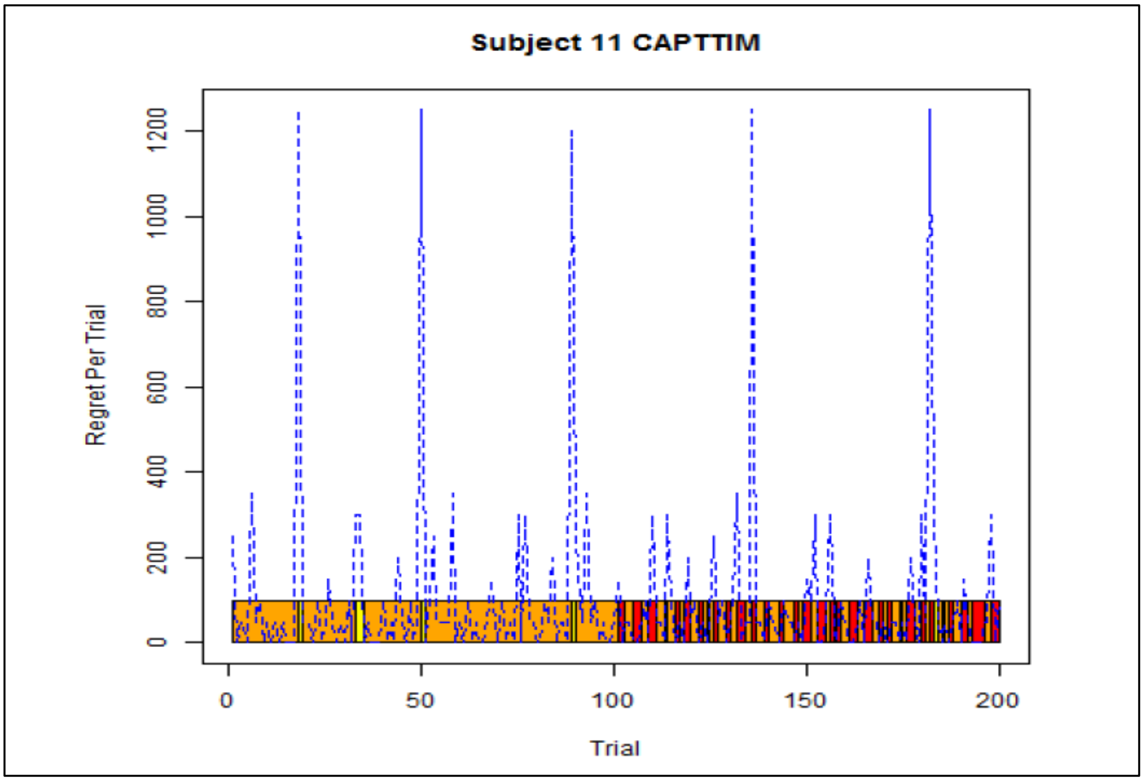
Figure 3. CAPTTIM Categories and Corresponding Cognitive State and Regret Information. (Source: Kennedy et al., 2015)

The CAPTTIM model has shown results that suggest we are able to (1) accurately classify a subject’s cognitive state and decision performance at the trial-by-trial level and (2) determine which subjects made the transition to the optimal decision path (Subject 14) and which subjects would benefit from individualized feedback (Subjects 11 and 33). (See Figures 4 through 6).



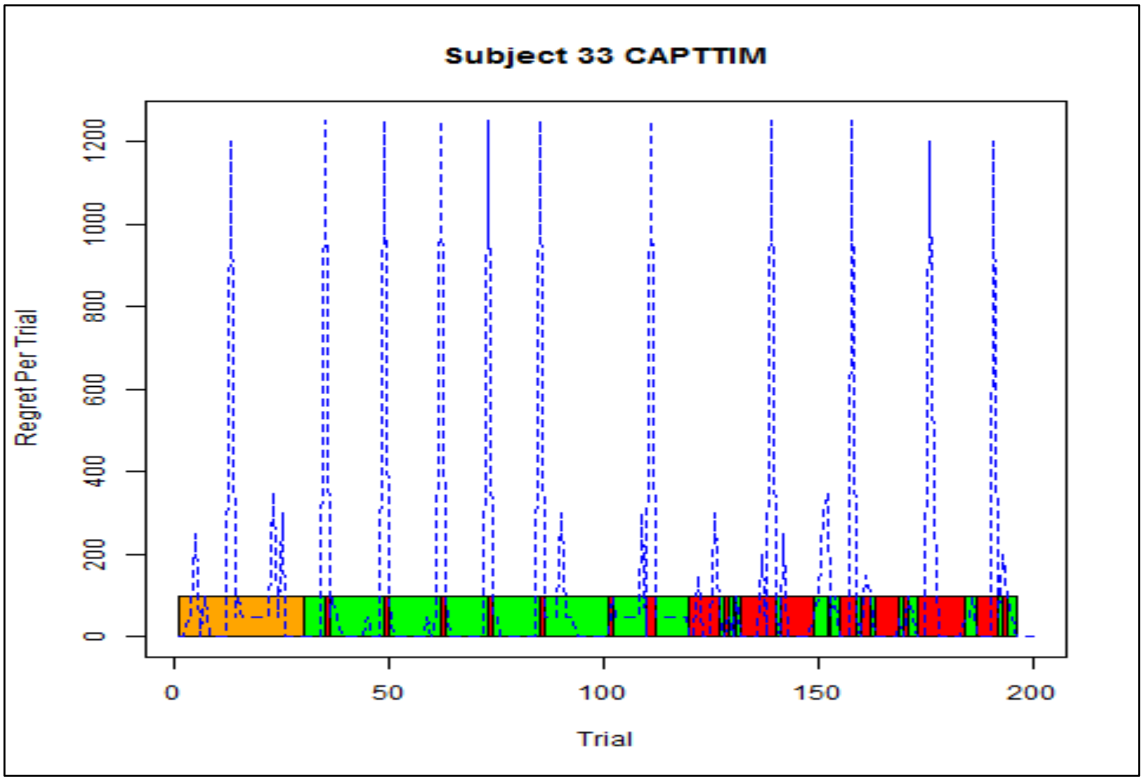
Subject 14 shows the ideal transition from exploration to optimal decision-making. Note: Yellow, orange, red, and green indicate CAPTTIM categorization for a given trial. Blue vertical spikes represent trials in which subjects received strong negative feedback.

Figure 4. Critz (2015) Subject 14 CAPTTIM Categorization of Decision Behavior at the Trial-by-Trial Level. (Source: Critz, 2015)



Subject 11 never quite figured out the task. Note: Yellow, orange, red, and green indicate CAPTTIM categorization for a given trial. Blue vertical spikes represent trials in which subjects received strong negative feedback.

Figure 5. Critz (2015) Subject 11 CAPTTIM Categorization of Decision Behavior at the Trial-by-Trial Level. (Source: Critz, 2015)



Subject 33 consistently exploited poor choices despite receiving strong negative feedback. Note: Yellow, orange, red, and green indicate CAPTTIM categorization for a given trial. Blue vertical spikes represent trials in which subjects received strong negative feedback. (Critz, 2015)

Figure 6. Critz (2015) Subject 33 CAPTTIM Categorization of Decision Behavior at the Trial-by-Trial Level. (Source: Critz, 2015)

As this thesis aims to demonstrate that decision performance can be improved using tailored messages when a subject's cognitive state is misaligned with decision performance, we must explore how to effectively communicate the need for a change in decision-making strategy; we need to be able to immediately and effectively communicate to a subject that their decision-making pattern is not optimal. That is, how and what do we communicate to a Subject 33 (depicted in Figure 6) that will cause decision performance to transition to optimal decisions such as portrayed by Subject 14 (depicted in Figure 4)?

C. DECISION-MAKING TRAINING INTERVENTION

Much of the challenge of the current CAPTTIM-based thesis was to convert the retrospective analysis of cognitive state and decision performance contained within the CAPTTIM model – and depicted in Figure 3 above – to a near real-time system. The current effort would only be fruitful when the real-time recognition of cognitive state and decision performance could be used to alert a subject to suboptimal performance and attempt to influence the decision-making strategy toward a preferred end state. Thus, one aspect of this thesis involved determining the type of feedback to give to subjects.

The type of feedback to give to subjects was guided by studying literature on other experience-based learners; i.e., language acquisition students. Most evident in the literature related to 'feedback to students' and/or 'intervention in education/training' was techniques used by second-language teachers and learners. In *Corrective Feedback and Learner Uptake,* the authors study when, how, and which learner's errors should be corrected (Lyster & Ranta, 1997). This information is pertinent to when, how and which subject's cognitive misalignments should be corrected or guided in our experiments. Of the six types of feedback discussed in a literature review, we surmise that, while effective, explicit correction could result in the decisions of our subjects being influenced too firmly toward the desired decision path.

We are studying whether a subject can learn through experience during execution of a task, not whether they understand their own particular reasoning behind the change in strategy. In our subjects, we are seeking self-repair, which “refers to a self-correction in response to the feedback when the latter does not already provide the correct form” (Lyster & Ranta, 1997, p. 50). We do not want to hand the subject the answer but rather guide their experience-based learning based upon our evaluation of their CAPTTIM classification (Red, Orange, Yellow, Green). Therefore, we crafted our feedback messages to subjects to be in the form of metalinguistic feedback, which *“*contains either comments, information, or questions related to the subject’s response, without explicitly providing the right answer” (Lyster & Ranta, 1997). The specific guidance offered to subjects based on their current CAPTTIM categorization will be detailed below.

D. THESIS MOTIVATION

Decision-making is what leaders do. As the decisions of military leaders become more and more complex, and have the potential for greater and greater impacts, it is imperative that we understand the process of decision-making and attempt to build training systems and techniques that develop leaders who understand how to tend toward optimal decisions. We want to evolve past using a single instructor’s best guess at whether a single trainee is making optimal decisions. This thesis extends upon past study on decision-making at NPS to attempt to capture the decision-maker’s cognitive state in real time, and further, influence sub-optimal decisions.

II. METHODS

The NPS Institutional Review Board approved our study to test whether CAPTTIM-oriented feedback could aid optimal decision-making; several methodological steps were completed to arrive at the final experiment. This section will first illustrate how previous work used a retrospective approach to identify a subject's cognitive state as exploring or exploiting the decision-making environment and to classify decisions as optimal or suboptimal decisions through the use of a quantitative metric of decision performance called regret. This previous work also showed that those two factors (cognitive state and decision performance) could be retrospectively combined to represent a subject's placement in CAPTTIM. Next, the methodological steps used in the current work to apply the CAPTTIM categorization in real time will be discussed. The final methodological steps were to use real time CAPTTIM categorization to provide timely and targeted feedback to subjects as they complete the Convoy Task. The Python executable code for the modified Convoy Task (with and without feedback windows) is available in Appendix B.

A. PREVIOUS WORK IN DEFINING COGNITIVE STATE AND REGRET

Previous work in decision-making at NPS has used the two factors of 'cognitive state' and 'decision performance' to classify the subject into one of four CAPTTIM, color-coded categories (Kennedy et al., 2015; Critz, 2015). The evolution of these factors and the demonstration that they can accurately categorize whether or not a subject's cognitive state is aligned or misaligned with observed decision performance will be used as foundation for application to real-time analysis of optimal decision-making.

1. Cognitive State: Exploration and Exploitation

Nesbitt et al. (2013) classified a subject's cognitive state by utilizing an exponentially weighted moving average (EWMA) of the latency between decision-making times.

An individual EWMA value is calculated as:

$$Z_i = \lambda X_i + (1 - \lambda) Z_{i-1}$$

where Z_i is the EWMA control statistic, λ is the weighted parameter, and X_i is the actual observed data value. The time between decisions was captured based on when a subject clicked on a route; using the computer's clock time we calculated the latency between clicks. Kennedy et al. (2015), showed that latency times would be exceptionally long after the subject experiences high damage, and that decision times after low damage would be relatively low. In order to determine whether a subject's latency time on a given trial was exceptionally long, a baseline latency time was established for each subject. Because previous work was completed retrospectively, all 200 trials were used to define the baseline as consisting of those latency times in which the subject received no to minimal friendly damage on the previous trial. Exploration thus was defined as a set of trials wherein the deviation between latency times was 2 SD or more greater than the baseline. Exploitation was defined as occurring on all other trials, i.e., trials in which the deviation between latency times was less than 2 SD above the baseline.

2. Regret as a Measure of Decision Performance

We use regret as the decision performance input to the real-time CAPTTIM category placement. Regret is the difference, in points, between an optimal decision and the subject's decision. Kennedy et al. (2015), Nesbitt, Kennedy, Alt & Fricker (2015) and, Critz (2015) all expanded from the IGT-based definition of regret in order to allow for more specificity in classifying users by CAPTTIM state. Because we know the payout of each route before the experiment begins, we also know, for any given trial, which route provides the best payout. Thus, regret can be calculated as the difference between the optimal score for a given trial and the score achieved by the subject's decision on that trial (Nesbitt et al., 2013).

Previous thesis work at NPS determined that the best method to delineate between a subjects' high or low regret is to compare the "process mean for a window of trials with the median of the process to determine whether it fell above or below the median. If the process mean was above the median, the subject was categorized as having high regret; if the process mean was below the median, the subject was categorized as having low regret" (Critz, 2015, p. 33). This information was derived by use of a statistical software program called R-studio and the use of built-in change point analysis tools.

Change point analysis is a method for determining whether a change has taken place in a set of values over time, and specifically upon which event or time that change happened. Software tools take a large set of data (whether non-normal distributions, ill-behaved, or data with outliers) and determine when significant changes occurred by noting a sudden change in direction of the cumulative sum (Taylor, 2000). Further, examining previous work and the establishment of the EWMA window we find that "the R package utilized in this analysis was the segment neighborhood algorithm which utilizes dynamic programming to calculate the optimal segmentation for m + 1 change points and reuses the data calculated for m change points" (Critz, 2015, p. 25). The algorithm examines an entire set and identifies where the set can be segmented to illustrate significant changes in value. As a subject's regret may change by many points at every decision this resulted in too many change points. Therefore, Critz (2015) specified a smaller number of changes (15) that still identified the subject's regret but did not display erratic, unreadable data.

B. PILOT TESTING

The modified Convoy Task code (Appendix B) was pilot tested to ensure it accurately reflected the foundational work creating and validating the CAPPTIM model (Nesbitt et al., 2013; Critz, 2015; Kennedy et al., 2015; Nesbitt et al., 2015). Pilot testing was conducted over two weeks and in two separate sessions. We used members of the thesis team who, while familiar with the overall construct of the experiment, did not have intimate knowledge of the modifications to the Convoy Task code and thus were able to provide usable feedback and data. We will highlight issues and resolutions of each pilot test period.

1. Pilot Testing: Correcting Modified Code for Cognitive State

Initial piloting runs exposed problems with converting the retrospective analysis of previous work to the real-time CAPTTIM assignment required of the hypothesis of this thesis. This piloting revealed the requirement to address discrepancies in the computation of a subject's baseline cognitive state. Recall that the explore/exploit cognitive state is assigned based upon the exponentially weighted moving average (EWMA) of the standard deviations of latency per trial as compared to the subject's baseline latency and SD thereof. Initial code modification of the Convoy Task stored the raw time between each decision as 'latency' and then queried this list to determine if the most recent decision was faster or slower than the overall average decision time. This method neglected two important contributing factors to properly computing a subject's cognitive state. First, because cognitive state characterization is based on variability in the SD of latencies, we need to establish the SD of a subject's baseline latency time and compare to that. Upon modification to the code, we used the first 50 trials of the Convoy Task to capture these baseline latency times and the SD associated with them. Latency times from only those decisions that did **not** result in exceptionally high Friendly Damage are stored and processed as the baseline. Second, as opposed to comparing the single most recent latency time to the baseline (or the overall average), the program is required to compare the standard deviation of the *ten* most recent trials to the standard deviation of the 50—good-decision—baseline trials. A threshold was applied to the SD of the EWMA of latency times in order to delineate between the cognitive states of exploration and exploitation. Based on extensive pilot testing, we calculated the SD of decision times and assigned to 'explore' or 'exploit' depending on whether that SD was above or below 1.5 times the standard deviation of baseline latency times. The actual number associated with the 'explore' or 'exploit' cognitive state was specific to each subject as the comparison was being made to his or her individual baseline. This extension of the foundational work of Nesbitt et al. (2013 and 2015) and Critz (2015) successfully incorporated the EWMA methodology to properly compare the standard deviations and determine whether the individual subject is making abnormally slower decisions than he or she normally would; in that case, for example, indicating an 'exploration' state.

2. Pilot Testing: Correcting Modified Code for Decision Performance

Pilot testing also allowed the team to discover aspects of the Python code used by Critz (2015) that did not directly translate to classifying a subject's decision performance (regret) in real time. Critz (2015) determined that a window of 15 trials worked well for retrospective categorization of regret. As current work did not require smooth transition curves to illustrate reinforcement learning, we chose to modify this window in the real-time analysis of regret. We were able to code a simple algorithm measuring subject performance and categorize regret according to the accepted EWMA model using only the previous 10 decisions. This modification allowed for more opportunities to observe variability in subject decision performance and (if subject is a member of the feedback group) to influence future decisions toward the optimal by displaying a message to guide decision-making strategy. We did maintain the same general model where high regret is defined as when the process mean for a certain number of trials is above the median for those same trials. However, we used a window of the last 10 decisions rather than 15 trials.

Further, while the code was originally written using the concept of 'gain' from the Iowa Gambling Task, the concept of regret—and its use as one variable of CAPTTIM classification—needed to be recognized and adapted as the opposite of gain in order to properly define whether the regret was "high" or "low." Initially we did not recognize that regret as we use it is the opposite of gain previously encoded in the Convoy Task, and thus we found that subjects' cognitive state was incorrectly assigned. Interestingly the regret assigned to a subject—high or low—was not exactly opposite of intended, which would have been the first assumption if ‘regret = ‘-gain’. Rather the inequalities used and the combination of the sliding window of trials resulted in unpredictable behavior but clearly improper assignment of CAPTTIM category. Once the pilot test revealed the incorrect assignment of CAPTTIM category, it was relatively simple to backtrack through the data and code by hand to realize that the inequalities in the code were reversed. This correction was made allowing us to use editable lists in the code to append the regret-per-trial value (called damage as per Nesbitt et al., 2015) and analyze the regret value of the previous 10 trials. The comparison of the median of the last 10 trials to the average as discussed above was relatively straightforward and all four categories of CAPTTIM (Red, Yellow, Orange, Green) were properly assigned to subjects during final pilot testing and on into experimentation.

Finally, an additional change we made from the original Convoy Task code with regard to regret was the automatic ‘red’ CAPTTIM categorization of those subjects who incurred extreme friendly damage after trial 100. Critz (2015) automatically assigned ‘high’ regret to subjects who incurred a ‘bad’ route after trial 100. We did not incorporate this classification into the Convoy Task, as it was our goal to show an ability to influence decision makers regardless of trial number. If we automatically placed subjects into a high regret state, we may have ended up displaying an improper message to a subject in the feedback group when another message may have been more appropriate given the regret state based purely on the mean/median comparison detailed above.

3. Pilot Testing: Capturing Data Outside of Established Change Points

As pilot testing continued, we realized that we did not have enough data during each subject’s run to confirm or reject the hypotheses regarding the proportion of trials in the green/red CAPTTIM classification. As mentioned above, Critz (2015) used a window of 15 trials in the change point analysis to determine when CAPTTIM classification occurs or changes. Thus, initially our Python code only captured the CAPTTIM classification every 15th trial after the baseline 50. This approach was acceptable to allow feedback to be issued to a subject in hopes of optimizing future decisions, but to analyze proportions after the completion of the experiment it was necessary to capture the cognitive state and regret data at each trial. The modification to the code was relatively minor (and is reflected in the final code used in the experiment as per Appendix B) but the correction to the design of the experiment was significant and allowed the team to move forward into experimentation confidently assured that enough data would be collected to compare between a control (no feedback) group and experimental (feedback) group.

4. Summary of Pilot Testing Changes

Overall, pilot testing illustrated four key changes to ensure the program used for this thesis captured and processed information effectively and categorized subjects into the validated CAPTTIM:

Incorporating the EWMA to analyze the SD of decision times and capture cognitive state vice simply comparing the latency times to a subject's average decision time.

1. Calculating the baseline latency time from the first 50 trials rather than retrospectively over the entire set of trials. Thus, in this study, the convoy task had 250 trials – the first 50 to acquire the baseline latency time and the remaining 200 for CAPTTIM assignment.

2. Correcting for an inaccurate assignment of high/low regret based on the subject's point gain as originally coded in Convoy Task; given the real-time nature of the data-capture in this thesis, the regret is captured in the same sliding window comparing the average of the last ten decisions to the median of the last ten, but we had initially not recognized the need to invert the properties for correct CAPTTIM assignment.

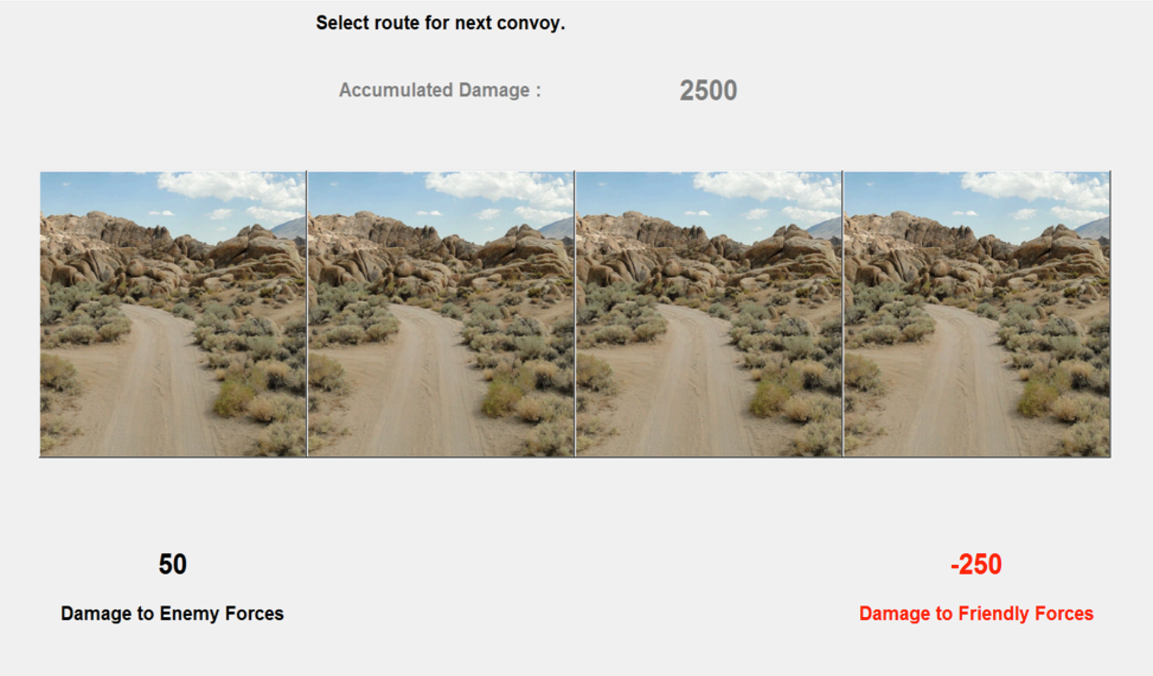
3. We discovered the need to capture the CAPTTIM category for each subject, on every trial, vice the set number of trials established by the change point analysis of Critz (2015). Ensuring the data was processed for every trial being one of the main goals of the thesis, this change – though relatively simple – was a key change exposed in the pilot tests.

C. PARTICIPANTS

All subjects were recruited from the student body of NPS. As such, all 34 were military officers, spanning all services: 14 U.S. Marine Corps, eight U.S. Army, eight U.S. Navy, and four U.S. Air Force. These subjects were randomly selected into two groups. There is no difference in demographic characteristics between the two groups (all p-values > 0.47). The control group and the feedback group both contained 17 subjects, 14 men and 3 women in each. The average age of the control group was 34.71 years (SD=3.64), and 32.53 years for the feedback group (SD=4.08 years). The control group had slightly more time in service: average of 13.47 (SD=4.56) years versus the feedback group's 10.06 years (SD=4.13 years). Despite the slight difference in years of service, the deployment record of the subjects within each group was the same: 14 members of each group had deployed to a combat zone and 3 had not, and the median of each group's members' return from the imminent danger pay deployment was 2013. The median rank was O-3 (lieutenant in the sea services, captain in the ground services and air force).

D. CONVOY TASK.

As detailed in Nesbitt et al. (2015) subjects saw four identical routes. (See Figure 7). Subjects were instructed that, over a pre-set number of trials, they choose which route to send convoys. Subjects will add to or subtract from their Accumulated Damage score by inflicting Enemy Damage or taking Friendly Damage respectively. Subjects were told during instructions that the pictures are identical. Their goal was to learn, by the experience of friendly and enemy damage at each trial, which routes achieve the maximum Accumulated Damage score.



Select route for next convoy.

Accumulated Damage : 2500

50

Damage to Enemy Forces

-250

Damage to Friendly Forces

The decision just executed by this subject has resulted in a gain of 50 damage points (Damage to Enemy Forces) and a loss of 250 damage points (Damage to Friendly Forces) for a net change to Accumulated Damage of -200 points.

Figure 7. Convoy Task Screen.

As can be seen in Appendix A, the routes have the same payout as the decks of cards in the original IGT (Bechara et al., 1994): routes 3 and 4 are considered good; routes 1 and 2 are considered bad. Participants receive immediate results of each trial by observing the Damage to Enemy Forces, Damage to Friendly Forces and Accumulated Damage score from the current decision.

E. FEEDBACK TO SUBJECTS

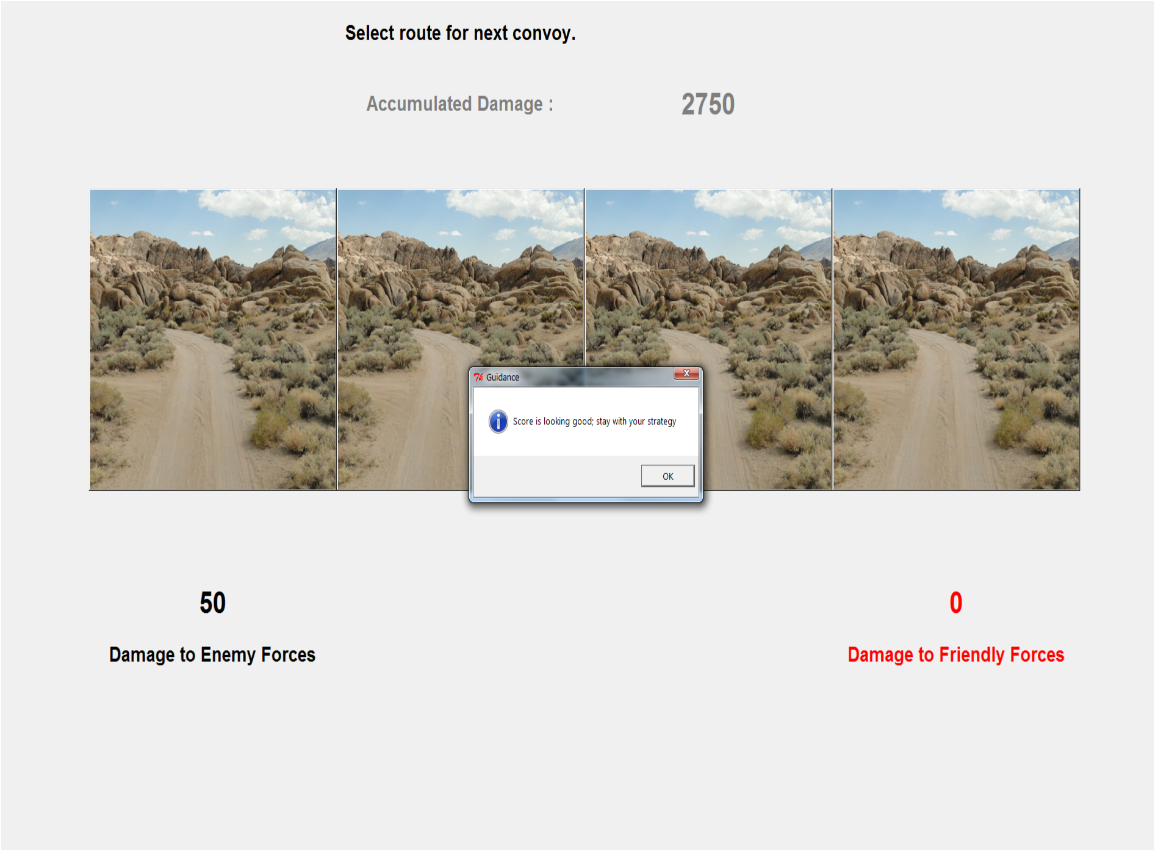
To examine whether messages to subjects can influence future decision-making toward optimal decisions—and whether there is a significant difference between those subjects and a control group that did not receive any feedback during execution of the task—we first must determine how to administer the feedback. We reviewed literature on feedback to trainees during execution of tasks and corrections made to students in second language learning (Archer, 2010, Chickering & Gamson, 1987, Lyster & Gamson, 1997) to determine the most acceptable method to offer input to subjects about their performance and decision making strategy. We had to decide carefully what information to provide the subjects to influence decision making without simply providing the exact proper strategy to succeed in maximizing score on the Convoy Task. We arrived at the messages corresponding to each CAPTTIM color category. (See Table 1). Also, a screenshot showing one of these four messages as seen by a subject is provided. (See Figures 8 and 9).

Further, we discussed when and how often to offer feedback. As we have already determined that the first 50 trials would be used to establish a subject's baseline latency time (the primary determinate of cognitive state), we continued the pattern and began the feedback to subjects after trial 50 and repeating every tenth trial. We demonstrate in the Results Section that the CAPTTIM categorization is knowable at every trial, allowing the messages in Table 1 to be displayed in pop-up windows when desired. Again, the executable Python code to view this computerized task is available in Appendix B.

Table 1. Messages Provided to Subjects in Feedback Group via Pop-up Windows

CAPTTIM Category	Message to subject in feedback group
Green (Exploit and low regret)	Score is looking good. Stay with your strategy
Yellow (Explore and high regret)	Score could be better; attend to friendly damage
Orange (Explore and low regret)	Score looking good, go ahead and make decisions quickly
Red (Exploit and high regret)	Score could be better, attend to friendly damage and try other routes.

Every 10 decisions after trial number 50 based on CAPTTIM category at that decision. Text in parentheses indicates the cognitive state and regret level associated with each CAPTTIM category



Select route for next convoy.

Accumulated Damage : 2750

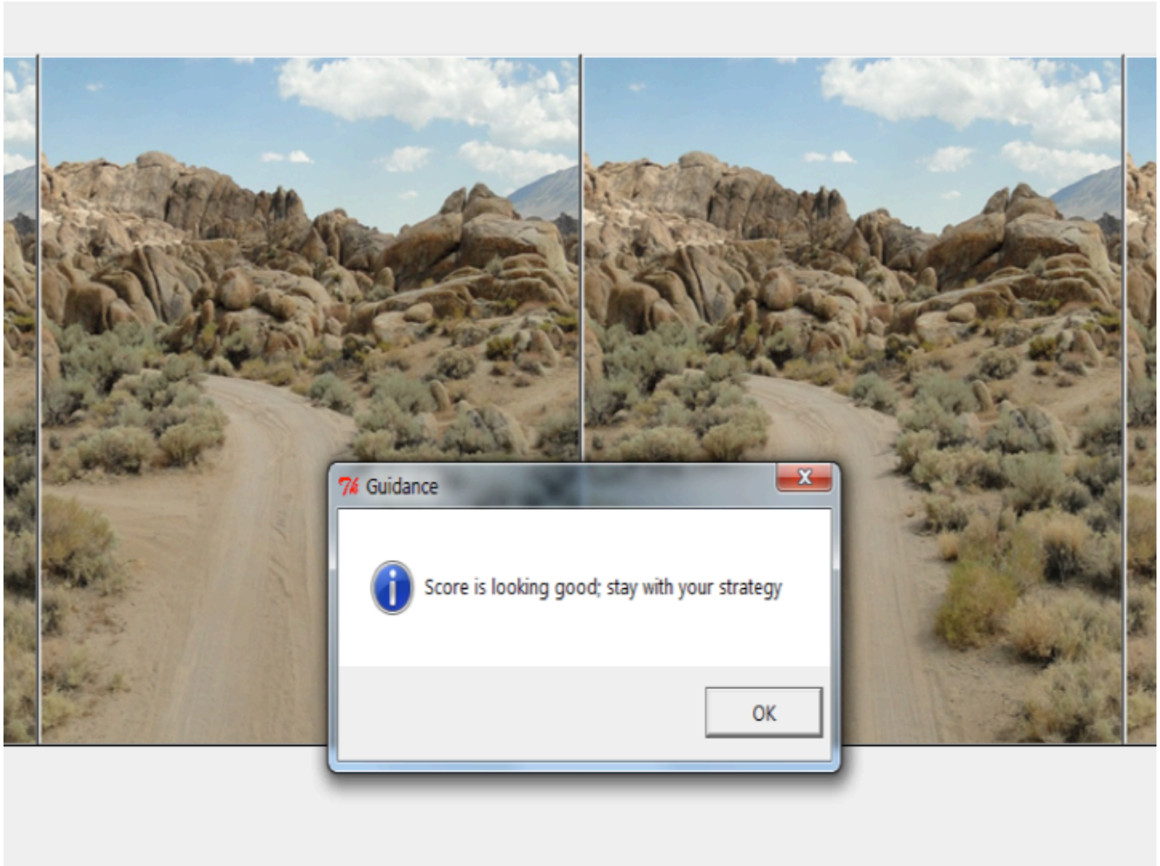
Score is looking good; stay with your strategy

50
Damage to Enemy Forces

0
Damage to Friendly Forces

.The strategy executed by this subject has resulted in CAPTTIM categorization of 'Green' and the resultant message of "Score is looking good; stay with your strategy" in the pop up window.

Figure 8. Convoy Task Screen Showing Feedback Pane



Score is looking good; stay with your strategy

Each of the four messages is displayed in a pop-up window with the same formatting, and requires the subject to click on the 'OK' button to continue the task. This subject is in the green CAPTTIM category thus is encouraged to stay with current decision-making strategy

Figure 9. A Closer Look at the Convoy Task Feedback Pane.

F. SURVEYS

We used surveys before and after the experiment to: 1) gather demographic factors that may have been relevant to statistical analysis and 2) collect strategies employed, and impressions of the experiment after completion.

1. Demographic Survey

The demographic survey included questions regarding branch of service, deployment history and general subject information such as age and rank. (See (Appendix C). This survey allowed us to verify the active duty military status of subjects and ensure results measured between the control and feedback groups are not due to other demographic characteristics

2. Post Task Survey

The post task survey queried subjects for qualitative input about their experience and decision-making strategy during the experiment. It also contained questions asking whether subjects changed their approach to decision-making during the task and if so, why. (See Appendix D).

G. PROCEDURES

This study was approved by NPS's Institutional Review Board. The overall concept of the experiment was to conduct the computerized Convoy Task on a single subject during a single visit to the lab. The experiment was designed to take less than one hour and was planned to take place during normal working hours at a time convenient to the individual volunteer subjects. Recruitment of subjects was conducted from among the student population of NPS by publishing a written advertisement on the school's intranet site where each student must read announcements once daily.

Once participants reported to the lab, an explanation of the general process was provided and the informed consent procedure was completed. If a subject consented to participate in an additional survey collecting data regarding head injuries, they completed The Ohio State University Traumatic Brain Injury (TBI) Identification short form. The data collected will not be discussed in this thesis as it is beyond the scope; the information was collected as part of a larger study. Whether or not a subject chose to participate in the head injury data collection, they completed a demographic survey as detailed above. The experimenter then randomly assigned subjects into the control group or the feedback group.

Eye-tracking hardware and software were calibrated to each individual to allow collection of gaze data during execution of the Convoy Task. The eye tracking software automatically generates two files that may be used to examine a subject's gaze point throughout the execution of the task and may then be analyzed to determine if there is correlation between designated factors (scores, proportion of time in each CAPTTIM category, etc.) and the subject's attention to data displayed on the screen. The eye tracking data was collected for a larger project and also will not be discussed here, as it is not within the scope of this thesis.

The experimenter used a script to explain the Convoy Task screen and task requirements to each subject in detail. Once the subject affirmed an understanding of the screen and the task, eye-track recording was begun and the subject was allowed to make decisions, uninterrupted, by using a mouse to click on a route. Each subject completed the Convoy Task by making 250 individual decisions to maximize a total score. If a subject was assigned to the feedback group, he or she received on-screen feedback via standard pop-up windows every 10 trials that offered guidance to the subject based upon their CAPTTIM categorization. If a subject was assigned to the control group, he or she received no on-screen feedback.

Finally, subjects answered the post task survey and the experimenter was available to answer questions about the study, its goals and potential uses of the results to develop training systems or techniques.

III. RESULTS

This section discusses the statistical results of the experiment and efforts to answer the research questions. In reviewing the results we will discuss subject-data preparation overall and whether the experiment was able to adequately answer the research questions and hypotheses in detail. The larger research questions to be addressed are 1. Whether cognitive state and decision performance (regret) data could be captured in real time while the subject completed the Convoy Task, and 2. Whether feedback offered to a subject based upon cognitive state and regret data would cause the subject to achieve better results (i.e., optimal decision making). The latter research question is divided into four hypotheses, which will be reviewed in detail and answered individually. Statistical methods and α-levels will be explained in conjunction with the specific hypotheses to which each applies.

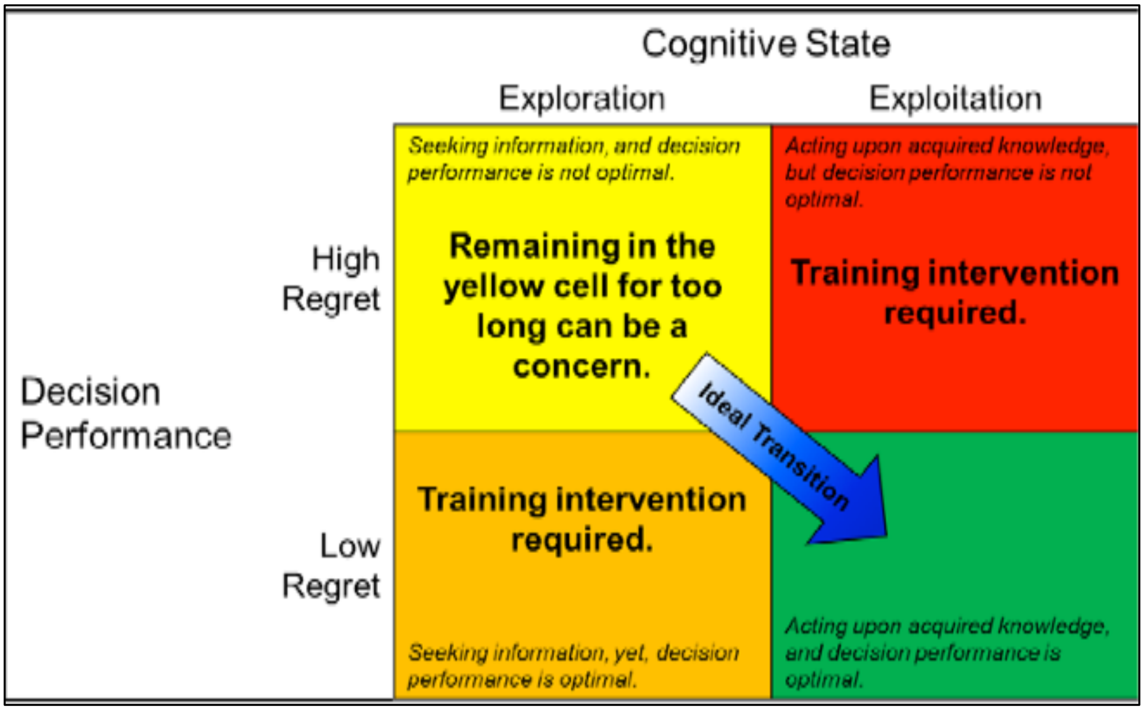
A. PRELIMINARY ANALYSES

Preliminary analyses revealed that there was no significant difference in demographic characteristics between the two groups detailed in the Participants section (all p-values > 0.47). Additionally, there were no significant differences in score performance on the Convoy Task by age, gender, military branch of service, or deployment history. For these demographic factors we used two-sample *t*-tests with a two-tailed, alpha level of .05 to compare means. When comparing for gender we find (*t*(34)=0.75, p=0.47) and find that mean scores are not significantly different by gender. Considering age, we separated the groups into old and young based upon the median age of all participants, 34 years. Eighteen subjects age 34 and older comprised the old group while the sixteen subjects aged 33 and younger comprised the young group. Using the same statistical procedure we find (*t*(34)=1.17, p=0.25), indicating that there is no significant difference in score by age. Similarly, when examining years of service we divided the groups based upon a more experienced service member (defined as the median—12 years—and greater) compared to a less experienced (11 years or less); the experienced group comprised of 18 subjects and less experienced counts 16 subjects. There is no difference in average scores between the two years-of-service groups (*t*(34)=183, p=.08). Thus, overall, we suggest that Convoy Task and CAPTTIM results cannot be explained by any potential difference in demographic characteristics between the control and feedback groups.

B. RESEARCH QUESTION 1: REAL-TIME DATA CAPTURE

In answer to the first research question, we found that the Python code in Appendix B was able to reliably capture subjects' decision-making data (cognitive state and decision performance) in real time.

The two factors are combined between each trial to result in an assigned CAPTTIM categorization as of that trial. If a subject is observed to be exploiting the environment (again this is when the standard deviation of current decision times is less than 1.5 times a subject's individual baseline standard deviation) but regret is high (i.e., not making optimal decisions) the subject's CAPTTIM categorization is red. Exploiting the decision-making environment with low regret earns a subject a green categorization. The Exploration cognitive states are similar: with high regret, yellow CAPPTIM; with low regret, orange CAPTTIM categorization. This dynamic can be concisely depicted by graphic. (See Figure 10.)


Cognitive State
Exploration
Exploitation
Decision Performance
High Regret
Low Regret
Seeking information, and decision performance is not optimal.
Remaining in the yellow cell for too long can be a concern.
Acting upon acquired knowledge, but decision performance is not optimal.
Training intervention required.
Ideal Transition
Training intervention required.
Seeking information, yet, decision performance is optimal.
Acting upon acquired knowledge, and decision performance is optimal.

Exploitation & High Regret = RED, Exploration & High Regret = YELLOW, Exploration & Low Regret = ORANGE, Exploitation & Low Regret = GREEN.

Figure 10. CAPTTIM Categorization States.
(Source: Kennedy et al., 2015)

Below is a sample of the data captured for each subject, and demonstrates that the desired data can be captured in real time, on a decision-by-decision basis and successfully categorizes a subject into the appropriate CAPTTIM category. (See Table 2). Note that Table 2 has been edited for space and that the selection of trials included are to demonstrate the effective capture of all CAPTTIM categories and not necessarily a complete record of the subject's consistent or overall performance. For example, it can be assumed that between trials 51 and 61 the subject remained in the red CAPTTIM category, and from 61 to 79 the subject was in the yellow category continually. But the overall capture of data and manipulation to CAPTTIM category on a decision-by-decision basis is demonstrated.

Table 2. Capture of CAPTTIM Real-time Data from Subject 211.

trial	routeSel	trialGain	trialLoss	Damage	latent	cogState	CAPTTIM
50	4	50	0	2450	0.645		
51	4	50	0	2500	0.946	Exploit	RED
59	4	50	0	2650	0.526	Exploit	RED
60	4	50	0	2700	0.546	Exploit	RED
61	4	50	0	2750	12.316	Explore	YELLOW
78	1	100	0	3400	3.046	Explore	YELLOW
79	1	100	250	3250	0.827	Explore	YELLOW
80	3	50	0	3300	2.269	Exploit	RED
84	3	50	0	3400	1.633	Exploit	RED
85	3	50	50	3400	0.927	Exploit	RED
86	3	50	50	3400	1.23	Exploit	GREEN
91	4	50	0	3550	7.04	Exploit	GREEN
92	4	50	0	3600	0.706	Exploit	GREEN
93	4	50	0	3650	0.647	Exploit	RED
94	4	50	0	3700	0.606	Exploit	RED
95	4	50	0	3750	0.566	Exploit	RED

Displayed in the table from left to right are the data points captured on each decision: The trial number (a count of the decisions which a subject has made), the route selected (numbered 1 – 4 from left to right as viewed on the experiment screen), the trial gain, a point value that the decision gained for the subject, the trial loss, a point value the subject lost for each decision (these values result in a net gain – can be positive or negative – for each decision), the running Damage score (a result of all of the previous net gains, which began as a value of 2000), the latent time between each decision in seconds, the 'explore' or 'exploit' cognitive state and the CAPTTIM color categorization.

We can also represent the percent of trials each subject in the control group spent in each CAPTTIM categorization (See Table 3). Also shown are the overall percent of time all subjects were in each of the four CAPTTIM categories.

Table 3. Control Group Subjects' CAPTTIM Breakdown for the Duration of the Experiment.

CONTROL GROUP				
SUBJECT	GREEN	YELLOW	ORANGE	RED
	Percent of trials in each color			
110	25	0	0	75
111	4.5	0	0	95.5
112	6.5	0	0	93.5
113	13	0	0	87
114	7.5	1.5	0	91
115	1	24.5	0.5	74
116	51	0	0	49
117	100	0	0	0
118	5.5	5	1.5	88
119	11.5	0	0	88.5
120	10	0	0	90
121	7	0	0	93
122	20	0	0	80
123	4	0	0	96
124	14	0	0	86
125	15.5	0	0	84.5
126	2	0	0	98
TOTAL (MEAN)	**17.52941176**	**1.823529412**	**0.117647059**	**80.52941176**
TOTAL (SD)	**24.3140074**	**5.973747715**	**0.376223494**	**23.78449507**

Category values are percentages are as percent of total number (250) of decisions. Also depicted (in bold at bottom of table) is the total percentage of decisions the group spent in the corresponding CAPTTIM color category

Table 4 represents the percent of trials each subject in the feedback group spent each CAPTTIM categorization. Also shown are the overall percent of time all subjects were in each of the 4 CAPTTIM categories.

Table 4. Feedback Group Subjects' CAPTTIM Breakdown for the Duration of the Experiment.

FEEDBACK GROUP				
SUBJECT	GREEN	YELLOW	ORANGE	RED
	Percent of trials in each color			
210	7.5	0	0	92.5
211	52.5	9	0	38.5
212	90	4.5	1.5	4
213	66	0	0	34
214	8	4.5	0.5	87
215	14	0	0	86
216	44	5	0	51
217	10.5	0	0	89.5
218	6	3.5	1.5	89
219	0.5	0	0	99.5
220	8	5	0	87
221	17.5	5	0	77.5
222	20	5	0	75
223	11.5	3	2	83.5
224	49	4.5	0	46.5
225	0.5	12.5	0.5	86.5
226	8.5	3.5	1.5	86.5
TOTAL (MEAN)	**24.35294118**	**3.823529412**	**0.441176471**	**71.38235294**
TOTAL (SD)	**26.10661118**	**3.381665531**	**0.704502327**	**26.55744329**

Category values are percentages are as percent of total number (250) of decisions. Also depicted (in bold at bottom of table) is the total percentage of decisions the group spent in the corresponding CAPTTIM color category

C. RESEARCH QUESTION 2: HYPOTHESES RELATIVE TO FEEDBACK PROVIDED TO SUBJECTS AIMED TOWARD OPTIMIZING DECISION MAKING

1. Data Preparation and Statistical Methods

Because the data did not conform to a Normal distribution curve, we used the nonparametric Wilcoxon Rank Sum test to test all hypotheses. A two-tailed alpha level of .05 was employed for all statistical tests. We found that there were two outliers, one each in the control and feedback group. We observed that in both the control and feedback groups there were subjects who achieved an unusually high (control group) and low (feedback group) score. The 7850-point total score of subject 117 in the control group is two standard deviations above the mean for the control group. Similarly, subject 219's score of -2300 is more than two standard deviations below than the mean for the feedback group. We will report results with this data included and also briefly discuss results with those subjects excluded from the calculations. Specific hypotheses relative to the subject performance were:

- $H0_1$: There is no difference in mean trial number of transition to the 'green' CAPTTIM classification between the feedback and no feedback groups.
- HA_1: Feedback group will demonstrate transition to the 'green' classification of CAPTTIM in fewer trials than subjects who receive no feedback.
- $H0_2$: There is no difference in mean total score between feedback and no-feedback groups.
- HA_2: Subjects who receive feedback during execution of the Convoy Task will accumulate a higher total score as compared to a no-feedback group.
- $H0_3$: The proportion of trials in the green classification will not be significantly different between feedback and no-feedback groups
- HA_3: Subjects who receive feedback during execution of the Convoy Task will achieve a greater proportion of trials in the green CAPTTIM classification than a no-feedback group.

- $H0_4$: The proportion of trials in the red classification will not be significantly different between feedback and no-feedback groups.
- HA_4: Subjects who receive feedback during execution of the Convoy Task will achieve a lesser proportion of trials in the red classification of the CAPTTIM model.

2. Results

Table 5 summarizes the overall results of each hypothesis detailed above and we discuss the detailed results of each conclusion below.

Table 5. Results of Hypotheses Including Test Statistics and P-values for Each Hypothesis.

Hypothesis description	**CONTROL mean (SD)**	**FEEDBACK mean (SD)**	**STATs**	**Conclusion**
H1: Feedback group will transition* earlier. * only 3/17 (c) and 5/17 (f) transition to green category.	115.3 (52.7)	136.6 (40.7)	N/A*	N/A*
H2: Average Score of feedback higher than control.	2782.35 (2556.91)	3617.65 (2457.07)	Z=1.206 p=0.228	Retain $H0_2$
H3: Proportion in Green of feedback group higher than control group.	.18 (0.24)	.24 (0.26)	Z=0.913 p=0.361	Retain $H0_3$
H4: Proportion in Red of feedback group lower than control group.	.80 (0.81)	.71 (0.27)	Z=1.433 p=0.153	Retain $H0_4$

a. Hypothesis 1

To address hypothesis 1 we defined a transition to the green CAPTTIM category as 20 or more consecutive trials in the green category. Due to the small sample size of subjects who effectively transitioned to the green category based on our definition (3 of 17 (~18%) control group and 5 of 17 (~29%) in feedback group) we did not statistically test this hypothesis. Excluding the single outlier in each group this number becomes even more difficult to analyze effectively with only 2 of 16 subjects from the control group transitioning, and 5 of 16 in the feedback group.

b. Hypothesis 2

Although the results regarding total score were not significant (Z=1.206, p=0.228), we observed that in both the control and feedback groups there were subjects who achieved an unusually high (control) and low (feedback) score. The 7850-point total score of subject 117 in the control group is two standard deviations above the mean for the control group. Similarly, subject 219's score of -2300 is more than two standard deviations below than the mean for the control group. Even while excluding these extreme values we achieve (*Z*=1.941, p=0.052). This value is still not statistically significant, but nearly so.

c. Hypotheses 3 and 4

The third and fourth hypotheses are related to each other as both pertain to the proportion of trials spent in the red and green CAPTTIM categories respectively. Again, while not statistically significant both results trend in the right direction: hypothesis 3 (Z=0.913, p=0.361), hypothesis 4 (Z=1.430, p=0.153). Subjects who received feedback during execution of the Convoy Task spent a lower proportion of decisions in the red category and a greater proportion of decisions in the green category than the control-group subjects. As outliers, subjects 117 (control group) and 219 (feedback group) had similar impacts to the mean proportion of decisions each group spend in the red or green CAPTTIM categories. Subject 117 was uncharacteristically in the green for 100% of the evaluated decisions. Conversely, subject 219 was in the red for 99.5% of the evaluated decisions. If we exclude the two outliers, we still fail to reject the null hypothesis for Hypothesis 3 regarding the proportion of trials in the green category, (but by a more slim margin) (*Z*=1.621, p=0.105). However, we do reject the null for Hypothesis 4 regarding the proportion of trials in the red category. (*Z*=2.186, p=0.029).

D. EXPLORATORY ANALYSIS

During review of the post-task surveys (Appendix D), subjects in both the control and feedback groups (four of 17 subjects and six of 17 subjects respectively) correctly identified the most dangerous route as route two. We sought to determine if correct identification of the most dangerous route was associated with optimal decision-making as defined by our hypotheses of a higher total damage score, greater proportion of decisions in the green CAPTTIM category or a lesser proportion of decisions in the red CAPTTIM category. Using a two-sample *t*-test to compare the means of the total damage scores and CAPTTIM proportions of the two groups (i.e., those that, post-task, correctly identified the most dangerous route and those that did not) we find that there is no difference in mean score between those that identified the most dangerous route (M=3330, SD=2261.78) and those who did not (M=3145.83, SD=2644.52): (*t*(34)=0.206, p=0.581). There also is no significant difference in the proportion of decisions that "dangerous route identifiers" (M=32.35, SD=29.08) and "non-identifiers" (M=16.18, SD=22.17) spent in the green CAPTTIM category (*t*(34)=-0.708, p=0.242) or the red category (*t*(34)=0.987, p=0.826). So while it is an interesting observation that some subjects correctly identify the most dangerous route, this sense does not necessarily contribute to optimal decision-making; just because a decision-maker can identify factors to avoid making the worst decision continuously apparently does not mean they apply an optimal strategy.

IV. DISCUSSION

Decision-making—understanding it, and improving the efficacy of it—continues to be a focus of effort throughout the DOD (Odierno & McHugh, 2015 U.S. Marine Corps, 2012). This thesis sought to further the efforts of previous work (Nesbitt et al., 2015; Kennedy et al., 2015; Critz, 2015) in capturing decision-making performance and increasing decision-making expertise. The primary goals of this thesis were to: (1) adapt a test of reinforcement learning (Convoy Task) and a validated model of decision-making classification (CAPTTIM) in order to categorize decision performance and cognitive state in real time and (2) given that effective real-time categorization, provide feedback to subjects whose performance was suboptimal in an effort to improve decision performance. The first goal was successfully accomplished. Results pertaining to the second goal showed trends toward effective influence of decision makers toward optimal decisions. Fine-tuning the model may allow significant results to be realized with the small sample size, but also given the trends of the results, increasing the sample size may improve the power of the statistical results. This final chapter discusses implications of the results, explores some limiting factors that were not explored statistically as part of the research and addresses areas of future work that should be explored.

A. IMPLICATIONS

The Convoy Task that was modified from Critz (2015) and Nesbitt et al. (2015) maintains a structure that requires subjects to be adaptive, mentally agile, and demonstrate reasoned decision-making skills. As recommended by Critz (2015), this thesis successfully modified CAPTTIM to act as a tutor to guide subjects toward optimal decision-making. Based on results from this thesis, the Convoy Task and CAPTTIM offer an enhanced capability to aid DOD research toward developing more effective decision makers.

The modification and employment of the Convoy Task and the effective, real time, CAPTTIM categorization may open further study into understanding and instructing optimal decision-making. The ability to categorize a decision maker’s cognitive state with their decision performance in real time could allow training systems to be designed to tailor training to the individual decision maker. CAPTTIM could be used to interrupt training that is trending toward suboptimal decision-making performance when a subject’s cognitive state is misaligned with their decision performance. Further, if future experiments demonstrate the ability to significantly change subject behavior during execution of a task, training exercises (whether task trainers, learning uptake exercises, etc.) may be designed with a built in mechanism to guide suboptimal performers by way of in-process feedback that takes into account the performer’s cognitive state; similar to the tailored guidance messages employed in this thesis.

Although this experiment consisted of a relatively simple task, the concept of categorizing both cognitive state and decision performance in real time can be expanded to existing training simulations that require multiple, complex, chained decisions where each decision can be influenced toward the optimal decision in order to maximize training value and improve the cognitive skills and effective decision making of small unit leaders. This idea is aligned with previous research in training effectiveness that suggests that “training interventions are required to improve teamwork skills, such as decision making, communications, shared situation awareness, leadership, and co-ordination, to ensure efficient team functioning. Such training results in more effective and efficient decision making accelerated proficiency and the development of expertise in individuals and teams” (Crichton & Flin, 2001, p. 259). This intervention is precisely the type of response that was attempted here; when a subject has made one, or a series, of incorrect decisions there is now a mechanism that can alert the subject to the suboptimal performance. More than just pointing out a wrong answer, this research categorized a subject’s *ability* to make correct decisions and a system’s or trainer’s ability act upon that categorization to help the subject make better decisions. Furthermore, the ability to guide a subject to understanding the problem at hand, and how to properly act within the decision environment—as opposed to merely pointing out the correct answer to a single task or situation—is crucial for learning and developing effective decision making expertise (Archer, 2010).

The ability to incorporate objective decision-making measures to any existing simulation, and demonstrate the optimal decision path (or the ability to correct deviation from it) may also reduce the time required in the trial and error phase of reinforcement learning, resulting in savings of time and money required to train military decision makers. Our results suggest that it is, in fact, possible to understand the decision-maker’s cognitive state in real time with simple behavioral measures. And, with further refinement to tailored feedback, this understanding will allow future leaders, instructors and trainers to leverage the power of this approach to improve the processes and methods used to understand effective decision-making.

B. LIMITATIONS

Observations during the collection and analysis of data for this thesis revealed potential limitations to the method and results presented above. Given the data-driven nature of the experiment and the neutrality of the software program capturing data, and classifying subjects in accordance with the CAPTTIM model, it is unlikely that these issues had an impact on results but should be discussed in order to improve future efforts using the same or similar methodology.

1. Feedback to Subjects

Post-task surveys, and comments volunteered by some subjects while being debriefed about the study, suggested that the messages offered to the feedback group might have caused confusion. (See Appendix D). Taken as a whole, these comments suggest that the timing and frequency of the feedback messages could be refined. For example, one subject reported that the comments provided in the pop-up windows were not timely enough to capture their immediate actions and their perceived performance at that moment in the experiment. The feedback windows, having been programmed to collect cognitive state, decision performance, and current CAPTTIM categorization every 10 trials and display it to the subject, may not account for subject strategy changes within this ten-decision window. This subject specifically noted that a feedback window directed them to "…attend to friendly damage and try other routes" (this indicates red CAPTTIM categorization). However, this subject, by their own recollection, had already made an adjustment to decision strategy and was beginning to make progress away from the red CAPTTIM category, but was then confused by the advice to try other routes. This same subject further stated that they considered the feedback windows might be experimentally designed as a distraction meant to be overcome by individual assessment of perceived performance, despite instructions that such feedback would be offered to guide subjects to optimal decision making. A closer inspection of the data showed that this subject's final Accumulated Damage score was an outlier beyond 2 SD below the mean score of the feedback group.

Conversely, another subject commented that the thesis might choose to "study how annoying those pop-up windows are." This subject had quickly recognized the long-term, overall, safest route and had adopted the optimal decision-making strategy to maximize Accumulated Damage score as per the instructions, and thus did not need the guidance to "…stay with your strategy" every tenth trial. This subject received the same message every tenth trial despite continued green CAPTTIM performance. In order to control possible confounding factors for this research, the conditions for eliminating the messages if a subject remained in green CAPTTIM category for a certain number of trials were not included. Dynamic intervention intervals should be added to the system for subsequent research in order to allow optimally aligned decision-making to continue uninterrupted.

Options for displaying the feedback to subjects every 15th trial, in line with the original change point value used in Critz (2015), or displaying feedback messages only when the CAPTTIM categorization changed were explored during the early phases of designing the present study. Ultimately, these approaches were dismissed in favor of a design that presented notification every tenth trial in order to maintain a uniform number of messages to subjects. Experimentally, this design provided a standard number of opportunities to influence performance and also eliminated a potentially confounding variable (variability in the frequency and timing of feedback messages) that would likely have threatened internal validity of this study.

2. Identification of Bad Routes Vice Optimal Decisions

Some subjects reported that they recognized the long-term danger of routes one and two, but also that these routes were safe for a set number of decisions before the imposition of high Friendly Damage. (See Appendix A). Thus, subjects stated that they were attempting to maximize score by selecting a known-unsafe route right up to the decision that would result in losing points but never figured out the pattern precisely enough to achieve a maximum score by this method; they attempted to “game the game,” but were rarely successful. Subjects who continue to make sub-optimal decisions, regardless of a score indicating otherwise, and messages indicating a flawed strategy, may be so focused on trying to game the system that they do not recognize their poor performance. Attentional tunneling, attending to a task or goal for longer than is optimal (Wickens & Alexander, 2009), is further evidence supporting the need to notify subjects of poor performance. Although the results of exploratory analysis suggested that subjects who identified the most dangerous route performed no differently than those who failed to do so, the use of such a strategy results in either inefficient allocation of cognitive resources during task completion, or a failure to recognize a more optimal strategy than their current decision-making approach. This situation can, however, be accounted for in the model. Critz (2015) modified the CAPTTIM model to automatically place the subject into the red category if they chose route two after trial 100. This methodology was not included in the real time categorization used here, as it was important to gather the subject's data and attempt to influence decisions. Because of this goal, automatically placing a subject in the red category would nullify some opportunities to evaluate cognitive state and regret to influence future decisions. Empirically based refinements to increase the sensitivity of the real-time data capture and analyses of CAPTTIM will enable finely tuning feedback messages and present the opportunity to address and avoid attentional tunneling. Advances in this area will increase the likelihood of keeping subjects from pursuing an ineffective strategy when it is evident that cognitive state is not aligned with performance.

C. FUTURE WORK

Based on the successful demonstration of real-time CAPTTIM categorization of decision making, and the trend towards influencing decision makers toward optimal decision making, future work should focus on: fine tuning the Convoy Task application (and incorporation of the CAPTTIM model therein) to ensure precise capture of CAPTTIM category; refining the feedback messages to more effectively influence decision performance; expanding the application of the Convoy Task and CAPTTIM to a population outside of NPS or a larger sample from the current population. Other areas of future work include using eye gaze patterns and individual difference factors such as head injury status to gain greater insights into why some subjects do not reach optimal decision-making. These areas are discussed below.

1. Refinement of CAPTTIM Coding and Feedback Messages

As mentioned, a limitation of the model as presently implemented is the rigidity of the feedback to subjects, and the confusion or frustration that this may cause to subjects. Ultimately the goal of ongoing research is to develop a system that is sensitive enough to detect when a subject is significantly off the optimal decision-making path and provide appropriate feedback to get them on the path at the “right” time. This goal can be accomplished through additional and/or more refined messages to each subject. For this research, messages were developed that correspond to each of the four CAPTTIM categories, each message attempting to influence subjects’ decision-making toward the green category. A future study should investigate the use of additional messages if a subject remains in the red CAPTTIM category after being alerted once or twice or three times that they should change strategy. Similarly the efficacy of displaying messages more frequently if a subject is in a suboptimal state (red, yellow or orange), and not at all if the subject has achieved the green category for 10 or more consecutive trials, should be investigated.

2. Expand Population of Interest

Although the sample for this study (drawn from current, active duty, NPS, officer-students) was uniquely suited to examine decision-making in a military themed task, an investigation including a broader demographic should be conducted. A larger, less homogeneous military population could include decision-makers of various ranks and experience, or from units and institutions not specifically focused on graduate level education. A typical, standing military unit is comprised of members of varied ranks and education levels, different decision-making requirements and different approaches to decision-making. As evidenced by age/military experience data from the sample in this study, the population at NPS has a considerable amount of decision-making experience, and brings the biases associated with experience to the task. It is possible that this experience caused decisions that were not anticipated in the coding for feedback messages. Thus, once the code is refined to account for differences in experience, the general approach used here could serve as the framework to examine the decision strategy of entry-level military members and compare those strategies to a group that has been educated and evaluated (possibly through real world experience) in crucial decision-making environments. Junior members, if they were on the optimal decision-making path (i.e., in the green CAPTTIM category) would be left to continue the immediate decision-making task. Senior, more experienced subjects, could be evaluated against a tighter standard of effectiveness; i.e., achieving the green category more quickly or requiring less focused feedback to adjust errors in strategy.

3. Expand to a More Complex Task

Subjects in our Convoy Task only had one decision to make repeatedly—which one of four routes to send your convoy. The CAPTTIM model for categorizing decision-making can be applied to each decision in a changing environment. Strategy, first-person-shooter, flight simulation and even board games require a series of decisions that are unique to the situation at hand but all may be categorized by the CAPTTIM method as the best possible decision at that time, or some suboptimal fraction of the best possible decision. Applying the evaluation and feedback approach demonstrated in this thesis to a more complex task may reveal facets of decision-making (and its effectiveness) that are not realized when a subject is faced with the same decision over and over again. While the Convoy Task—and the underlying IGT—have been shown repeatedly to effectively capture decision-making performance, a deviation from this singly focused task would be illuminating.

4. Use of Eye Gaze Data

As mentioned in the Procedures Section, eye-tracking cameras were used to capture the gaze point on the task screen of each subject. These data were beyond the scope of this thesis. However, as the data is collected and preserved it could be examined retrospectively to determine if there is a difference in gaze points between high and low scoring subjects or between feedback and control subjects. It may be informative to know if the subjects in the feedback group really spent any significant time reading the messages that were displayed to them regarding adjusting strategy or if they allocated more attention to the most relevant piece of information, Damage to Friendly Forces. It also would be informative to see if those subjects who attempted to ‘game the game” were less likely to attend to the Accumulated Damage score.

5. The Role of Head Injury Incidence in Explaining Some of the Large Variability in Convoy Task Scores

Similarly, self-reported head injury incidence was collected but the analysis of this data was outside the scope of this thesis. Head injury incidence can be used as an indicator of TBI. Future effort may be used to examine this data and whether the role of head injury incidence explains some of the large variability in convoy task performance, or whether those with a history of frequent and/or severe head injuries would differentially benefit from feedback than those without such a history. We balanced those with varying degrees of head injury incidence between the two groups, but future studies may block all subjects with indicators of TBI into one group to see if the feedback has any effect given the history of brain injury. Previous results suggest that those with self-reported TBI show unusual decision performance patterns (Kennedy, Adamson, Huston & Nesbitt, 2015).

D. CONCLUSION

Decision-making is an everyday task that takes on greater significance to military professionals, first responders, or others faced with outsized impacts of a given set of decisions. Future U.S. military capability will be evaluated on the ability of military members' effective, agile, adaptive, and innovative decision-making (Odierno & McHugh, 2015). Rather than the acquisition of material solutions, development of personnel lends gravity to the research conducted here. More than just a necessity driven by budget cuts, advances in technology and application of innovative methods of simulated and virtual-environment training is an opportunity to improve performance of the modern military. The tasks and situations faced by every military member call for advanced understanding of the individual's decision-making capability, and development of the same in a manner never expected of previous generations.

We have shown that it is possible to capture the cognitive state and decision performance of subjects in real time. There are myriad factors that drive an individual’s decision-making strategy which require further exploration. However, by continuing to explore this process, this research moves closer to effective development of continuous, objective, measures and analysis capability for long-term tracking of decision-making skills. Understanding and influencing military decision-making is astutely paired with advances in virtual environments and simulated training. Investigating, developing and applying innovative approaches to training and education and incorporating the evaluation and intervention strategy applied here increases the potential to effectively train optimal decision-making in less time.

APPENDIX A. IGT PAYOUT

IGT Payout Schedule			
Deck A	Deck B	Deck C	Deck D
-150	100	50	50
-250	100	0	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	-1150	0	50
100	100	0	-200
-150	100	50	50
-250	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50

Convoy Task Payout Schedule			
Route 1	Route 2	Route 3	Route 4
-150	100	50	50
-250	100	0	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	-1150	0	50
100	100	0	-200
-150	100	50	50
-250	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50

IGT Payout Schedule (cont’d)			
Deck A	Deck B	Deck C	Deck D
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50

Convoy Task Payout Schedule (cont’d)			
Route 1	Route 2	Route 3	Route 4
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50

IGT Payout Schedule (cont'd) (3)

Deck 1	Deck 2	Deck 3	Deck 4
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50

Convoy Task Payout Schedule (cont'd) (3)

Route 1	Route 2	Route 3	Route 4
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50

IGT Payout Schedule (cont'd) (4)

Deck 1	Deck 2	Deck 3	Deck 4
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50

Convoy Task Payout Schedule (cont'd) (4)

Route 1	Route 2	Route 3	Route 4
100	100	0	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50

IGT Payout Schedule (cont'd) (5)			
Deck 1	**Deck 2**	**Deck 3**	**Deck 4**
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50

Convoy Task Payout Schedule (cont'd) (5)			
Route 1	**Route 2**	**Route 3**	**Route 4**
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200
-250	100	50	50

IGT Payout Schedule (cont’d) (6)

Deck 1	Deck 2	Deck 3	Deck 4
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200

Convoy Task Payout Schedule (cont’d) (6)

Route 1	Route 2	Route 3	Route 4
100	100	50	50
100	100	0	50
-50	100	50	50
100	100	0	50
-200	100	50	50
100	100	0	50
-100	100	50	50
100	-1150	0	50
-150	100	0	-200

APPENDIX B. CONVOY TASK CODE

A. CONTROL GROUP CODE

```
#  Optimal Decision Making Demonstration

#  Military Wargaming, Convoy Route Selection

#

#  Multi Arm Bandit (n=4)

#  In support of TRAC Project Code 638

#  author: Peter Nesbitt and Cardy Moten III, TRAC-MTRY

#  peter.nesbitt@us.army.mil or cardy.moten3.mil@mail.mil

#  addition/modification of COGNITIVE STATE and REGRET

#  Travis Carlson, MOVES, NPS

#-------------------------------------------#

#       IMPORTS                #

#-------------------------------------------#


from random import *

import random

import numpy as np

import time

from time import localtime, strftime

from math import *

import Tkinter

from Tkinter import *
```

```
import tkMessageBox
import tkFont
import Tkinter as tk
from PIL import Image, ImageTk
import winsound
import csv
import array
from datetime import datetime
import calendar
#-------------------------------------------------------#
#     FUNCTIONS AND CLASSES        #
#-------------------------------------------------------#

class Player:
  def __init__(self,onhand,plays):
    self.oh = onhand
    self.p = plays

class Bandit:
  def __init__(self,I_gain,I_loss,I_payoff):
    self.gain = I_gain # dictionary of initial bandit parameters
    self.loss = I_loss
    self.po = I_payoff # total earned for that machine
```

```
class Click:
  def __init__(self,xcoord,ycoord):
    self.x = xcoord # x for every click on canvas
    self.y = ycoord # y for every click on canvas

class Control:
   def __init__(self,l_routeUse):
    self.route = l_routeUse
    self.playlimit= 250

class DecideTime:
  def __init__(self):
    self.start = time.time() # time since last decision

class Application(Frame):
  def __init__(self, master=None):
    Frame.__init__(self, master)
    self.pack()
    self.buildFrame()
    self.remaining = 0

  def restart(self, remaining = None):
    if remaining is not None:
      self.remaining = remaining
      if self.remaining <= 0:
        self.hurry.configure(text=“time’s up!”)
      else:
        self.hurry.configure(text=“%d” % self.remaining)
        self.remaining = self.remaining - 1
        self.after(1000, self.countdown)

    def buildFrame(self, remaining = None):
      # self.buildFrame2()
      self.customFont1 = tkFont.Font(family=“Arial Bold,” size=30)
      self.customFont2 = tkFont.Font(family=“Arial Bold,” size=20)
      self.customFont3 = tkFont.Font(family=“Arial Bold,” size=10)
      self.customFont4 = tkFont.Font(family=“Arial Bold,” size=20)

      topLabel = Label(self,text=“Select route for next convoy.,”
font=self.customFont2).grid(row=0,column=2, pady=25)
      #topLabel = Label(self,text=“Select route for next convoy.,”
font=self.customFont2).grid(row=0,column=1, columnspan=3, pady=25)
      #Label(self,text= ““).grid(row=5,column=2)
      Label(self,text= “Damage to Enemy Forces,”
font=self.customFont2).grid(row=30,column=1,pady=25)

      #Label(self,text=0,                                   fg=“black,”
font=self.customFont1).grid(row=7,column=2)
```

```
        l1 = Label(self,textvariable=v_gain, fg=“black,”
font=self.customFont1).grid(row=25,column=1, pady = 0)

        Label(self,text= “Damage to Friendly Forces,”fg=“red,”
font=self.customFont2).grid(row=30,column=4)

        l2 = Label(self,textvariable=v_loss, fg=“red,”
font=self.customFont1).grid(row=25,column=4, pady = 0)


        b_1 = Button(self,command=bdt1,bg=‘white’)

        b_2 = Button(self,command=bdt2,bg=‘white’)

        b_3 = Button(self,command=bdt3,bg=‘white’)

        b_4 = Button(self,command=bdt4,bg=‘white’)


        self.photo1=ImageTk.PhotoImage(file=“Picture2.png”)

        self.photo2=ImageTk.PhotoImage(file=“Picture2.png”)

        self.photo3=ImageTk.PhotoImage(file=“Picture2.png”)

        self.photo4=ImageTk.PhotoImage(file=“Picture2.png”)


        b_1.config(image=self.photo1, width=“400,”height=“400”)

        b_2.config(image=self.photo2, width=“400,”height=“400”)

        b_3.config(image=self.photo3, width=“400,”height=“400”)

        b_4.config(image=self.photo4, width=“400,”height=“400”)


        b_1.grid(row=6, column=1)

        b_2.grid(row=6, column=2)
        b_3.grid(row=6, column=3)
        b_4.grid(row=6, column=4)

        l30 = Label(self, text=,”“ fg=“gray50,” font=self.customFont3,anchor=E).grid(row=3,column=1, pady=20)
        l35 = Label(self, text=,”“ fg=“gray50,” font=self.customFont3,anchor=E).grid(row=7,column=2, pady=20)
        l36 = Label(self, text=,”“ fg=“gray50,” font=self.customFont3,anchor=E).grid(row=8,column=2, pady=20)
        l3 = Label(self,text=“Accumulated Damage :,”fg=“gray50,” font=self.customFont2,anchor=E).grid(row=1,column=2, pady=20)
        l4 = Label(self,textvariable=v_onhand, fg=‘gray50’, font=self.customFont1).grid(row=1,column=3, pady=25)
        #l5 = Label(self,text=“(Positive number is good),”fg=“gray50,” font=self.customFont2,anchor=E).grid(row=1,column=4)



    def callback(e):
      click.x = e.x
      click.y = e.y
      # print “clicked at,” e.x, e.y

    def WriteToFile(listArray, subName):
      with open(subName, ‘wb’) as csvfile:
        w = csv.writer(csvfile)
```

```
  w.writerow([‘trial’] + [‘routeSel’] + [‘trialGain’] + [‘trialLoss’] +[‘Damage’] + [‘x’] + [‘y’] + [‘latent’] + [‘unixTime’]+ [‘machTime’] + [‘cogState’] +[‘CAPTTIM’])
  for e in listArray:   # for every trial data array,
   w.writerow(e)    # write it to file


def readFromFile():
 with open(‘TDC.csv’, ‘rb’) as f:
  reader = csv.reader(f)
  for row in reader:
   latentlistR = row[7]
   choiceRead = row[1]


   print “route: ,” choiceRead, “latency: ,” latentlistR
   #print row[3]
#-------------------------------------------#
#      GLOBAL CONSTANTS       #
#-------------------------------------------#


runData = [] # storage tuple to temp store data
latentList = [] #store latency times
latentListR = []
latentList50 = [] #have to use something different for first 50 becuase we need the whole latentList later
avglatentList = [] #store EWMA latency times
```

```
    avgLatencyTime = 0
    latencyLambda = 0.9 #EWMA lambda parameter
    sdLatencyTime = 0.0 #standard deviation of latency time
    cogState = ‘test’ #Cognitive State string
    baseLineLatency = 0
    #List of intervention messages
    messageList = [“Score could be better; attend to friendly damage,”
        “Score could be better; attend to friendly damage and try other routes,”
        “Score is looking good; go ahead and make decisions quickly,”
        “Score is looking good; stay with your strategy”]
    CAPTTIM = ‘ ‘
    gainList = [] #Capture absolute regret
    medGain = [] #Capture median regret values
    subName = strftime(“%Y %b %d %a %H %M Mil MultiArmBandit.csv,” localtime()) # time as file name
    root = Tkinter.Tk( )
    # Player parameters
    x = Player(2000,0) # instantiate player object onhand,plays
    irouteUse= {}
    irouteUse[1]= 0
    irouteUse[2]= 0
    irouteUse[3]= 0
    irouteUse[4]= 0
```

```
game = Control(irouteUse)

time_to_decide = DecideTime()


click = Click(0,0)

v_onhand = DoubleVar() # instantiate running total onhand

v_onhand.set(x.oh) # running total from all machines

Dcolor = ‘black’

v_gain = DoubleVar()

v_gain.set(0) # running total from all machines

v_loss = DoubleVar()

v_loss.set(0) # running total from all machines

v_plays = IntVar()

#v_plays.set(game.playlimit-x.p)

v_plays.set(x.p)


v_bdt1 = DoubleVar() # last payoff value for machine 1

v_bdt2 = DoubleVar()

v_bdt3 = DoubleVar()

v_bdt4 = DoubleVar()


v_gain1 = DoubleVar()

v_gain2 = DoubleVar()

v_gain3 = DoubleVar()

v_gain4 = DoubleVar()
```

```
        # Bandit parameters held in a dictionary

        igain= {}

        igain[1]= 100 #bandit 1: n1,p1,n2,p2

        igain[2]= 100

        igain[3]= 50

        igain[4]= 50


        iloss= {}

        iloss[1]=    [-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-
150,

            0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,

            -150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,

            -300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,

            -200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,

            -250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,

            -350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,

            0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-
150,0,0

            -350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,

            0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,

            -150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,

            -300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,

            -200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,
            -250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,
            -350,-250,0,-200,0,-300,0,-150,
            0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-
150,0,0
            -350,-250,0,-200,0,-300,0,-150,0,0,-350,-250]

        iloss[2]=                    [-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-
1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,
            -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0]

        iloss[3]=     [-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-
50,0,-50,0,
            -50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-
50,0,0,
            -50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-
50,
            0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-
50,0,
            0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,
            -50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,
            -50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,
            -50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,
            -50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,
            0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,
            -50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,
            -50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,
            -50,-50,0,-50,0,-50,0,-50,0]

        iloss[4]=                    [-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-
250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
            -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0]
        print ‘Deck A is’,len(iloss[1]),’cards long.’
        print ‘Deck B is’,len(iloss[2]),’cards long.’

        print ‘Deck C is’,len(iloss[3]),’cards long.’

        print ‘Deck D is’,len(iloss[4]),’cards long.’


        ipayoff= {}

        ipayoff[1]= 0

        ipayoff[2]= 0

        ipayoff[3]= 0

        ipayoff[4]= 0


        b=Bandit(igain,iloss,ipayoff)


        v_bdt1.set(b.po[1])

        v_bdt2.set(b.po[2])

        v_bdt3.set(b.po[3])

        v_bdt4.set(b.po[4])


        v_gain1.set(0)

        v_gain2.set(0)

        v_gain3.set(0)

        v_gain4.set(0)


        def refresh():

          app.mainloop()
```

```
def bdt1():
  machine= 1
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)

def bdt2():
  machine= 2
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)

def bdt3():
  machine= 3
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)

def bdt4():
  machine= 4
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)
```

```
#Display a modal pop-up info box with the supplied message string
def displayDialog(message):
  tkMessageBox.showinfo(“Guidance,” message)


def getGain(gain,loss,mach):
  CAPTTIM = ‘‘
  cogState = ‘‘
  gainP = gain
  lossP = -1*loss.pop()
  gain = gainP - lossP
  latent = time.time() - time_to_decide.start
  time_to_decide.start = time.time()
  game.route[mach] += 1
  b.po[mach] = b.po[mach] + gainP + lossP # update earning by machine
  x.oh = x.oh + gain  # update earnings total, subtracting any cost to play
  if x.oh < 0:
    Dcolor = ‘red’
  else:
    Dcolor = ‘black’
  v_onhand.set(x.oh)
  x.p = x.p + 1          # update times game played
  dt = datetime.now()
  machTime= dt.time()
```

```
unixTime = calendar.timegm(dt.utctimetuple())


if x.p <= 50:
 selData=[x.p, mach, gainP,lossP,x.oh, click.x, click.y, latent,unixTime,machTime, CAPTTIM]
 runData.append(selData)    # store data


##################################
#   Cognitive State    #
##################################


 if x.p<=2:
  avgLatencyTime = latent
  avglatentList.append(latent) #Store Average Latency Time
  latentList.append(latent)
  latentList50.append(latent)
 else:
  if gain >= 0: ##only append the latency time to the list if the choice is not ‘bad’
   latentList.append(latent)
   latentList50.append(latent)
  #Compute EWMA Latency from Nesbitt Understanding Optimal Decision Making
   avgLatencyTime = latencyLambda*latentList[len(latentList)-1] + (1-latencyLambda)*avglatentList[len(avglatentList)-2]
```

```
                #Store Average Latency Time of these GOOD decisions
                    avglatentList.append(avgLatencyTime)
                else:
                    avgLatencyTime = latencyLambda*latentList[len(latentList)-1] + (1-latencyLambda)*avglatentList[len(avglatentList)-2]
                    #Still computing the average latency time, just not appending it to the list when subject takes a hit during first 50 trials
                    print “bad choice, latency not added to avgLatentList”


            baseLineLatency = np.mean(latentList50)


        else:
            latentList.append(latent) #still have to capture all the raw times? We should be using EWMA for Exp/Exp
            baseLineLatency = np.mean(latentList50) ##there’s nothing added to latentList after trial 50, so baseLineLatency stays the same
            STDofBaseLineLatency = np.std(latentList50) #Compute the standard deviation of the latency time
            avgLatencyTime = latencyLambda*latentList[len(latentList)-1] + (1-latencyLambda)*avglatentList[len(avglatentList)-2]
            avglatentList.append(avgLatencyTime)
            print “SD of baseline,” STDofBaseLineLatency
            #get the mean of the last 10 latency values from the overall list
            Last10AvgLatencies = avglatentList[len(avglatentList)-10:len(avglatentList)]
```

```
  STDLast10 = np.std(Last10AvgLatencies)
  print “STD of Last 10 trials: ,” STDLast10


  if STDLast10 <= 1.5*STDofBaseLineLatency:
    cogState = ‘Exploit’
  elif STDLast10 > 1.5*STDofBaseLineLatency:
    cogState = ‘Explore’
  print “CogState is: %s” %cogState


##################################
#    Regret      #
##################################


gainList.append(gain)
regret = -gain
for check in range(50,game.playlimit,10):
  checkLast10 = gainList[len(gainList)-10:len(gainList)] #take the most recent 10 gain values from the overall list
  checkLast15 = gainList[len(gainList)-15:len(gainList)] #take the most recent 15 gain values from the overall list
  averageLast10 = np.average(checkLast10)
  medianLast10 = np.median(checkLast10) #the median of the above list to compare against most recent trial
  medianLast15 = np.median(checkLast15)
  averageLast15 = np.average(checkLast15)


  if x.p == check:
    print “The last 10 gains: ,” checkLast10
    print “median of last 10 trials: ,” medianLast10
    print “average of last 10 trials ,” averageLast10


    if averageLast10 < medianLast10 and cogState == ‘Explore’:
     CAPTTIM = “YELLOW” #the inequality above “ave > median” is the definintion of gain (note line 381 that regret is opposite of gain)
     #displayDialog(messageList[0])
    elif averageLast10 < medianLast10 and cogState == ‘Exploit’:
     CAPTTIM = “RED”
     #displayDialog(messageList[1])
    elif averageLast10 >= medianLast10 and cogState == ‘Explore’:
     CAPTTIM = “ORANGE”
     #displayDialog(messageList[2])
    elif averageLast10 >= medianLast10 and cogState == ‘Exploit’:
     CAPTTIM = “GREEN”
     #displayDialog(messageList[3])
    print “CAPTTIM,” CAPTTIM


## Compute CAPTTIM for every trial and append it to the selection data for later evaluation of proportion of time in R/Y/O/G
```

```
if x.p>50:
  if averageLast10 < medianLast10 and cogState == ‘Explore’:
    CAPTTIM = “YELLOW”
  elif averageLast10 < medianLast10 and cogState == ‘Exploit’:
    CAPTTIM = “RED”
  elif averageLast10 >= medianLast10 and cogState == ‘Explore’:
    CAPTTIM = “ORANGE”
  elif averageLast10 >= medianLast10 and cogState == ‘Exploit’:
    CAPTTIM = “GREEN”
  selData=[x.p, mach, gainP,lossP,x.oh, click.x, click.y,
latent,unixTime,machTime,cogState,CAPTTIM]
  runData.append(selData)


v_gain.set(0)
v_loss.set(0)
if mach == 1:
  v_bdt1.set(b.po[mach])
  v_gain1.set(gain)
if mach == 2:
  v_bdt2.set(b.po[mach])
  v_gain2.set(gain)
if mach == 3:
  v_bdt3.set(b.po[mach])
  v_gain3.set(gain)
  if mach == 4:
    v_bdt4.set(b.po[mach])
    v_gain4.set(gain)

  v_plays.set(x.p)
  v_gain.set(gainP)
  v_loss.set(-1*lossP)

  if x.p >= game.playlimit:
    print (“\n\nPLAY LIMIT MET\n\n”)
    WriteToFile(runData,subName)
    print (‘shut down’)
    root.quit()\

if __name__ == “__main__”:
  app = Application(master=root)
  app.master.title(“Route Selection and Battle Damage Tool”)
  app.master.minsize(1000,400)
  root.bind(“<1>,” callback)
  app.mainloop()
  root.destroy()
```

B. FEEDBACK GROUP CODE

```
# Optimal Decision Making Demonstration
#  Military Wargaming, Convoy Route Selection

#

#  Multi Arm Bandit (n=4)

#  In support of TRAC Project Code 638

#  author: Peter Nesbitt and Cardy Moten III, TRAC-MTRY

#  peter.nesbitt@us.army.mil or cardy.moten3.mil@mail.mil

#  addition/modification of COGNITIVE STATE and REGRET

#  Travis Carlson, MOVES, NPS

#-------------------------------------------#

#      IMPORTS           #

#-------------------------------------------#

from random import *

import random

import numpy as np

import time

from time import localtime, strftime

from math import *

import Tkinter

from Tkinter import *

import tkMessageBox

import tkFont

import Tkinter as tk

from PIL import Image, ImageTk

import winsound
import csv
import array
from datetime import datetime
import calendar
#------------------------------------------#
#    FUNCTIONS AND CLASSES      #
#------------------------------------------#


class Player:
  def __init__(self,onhand,plays):
    self.oh = onhand
    self.p = plays


class Bandit:
  def __init__(self,I_gain,I_loss,I_payoff):
    self.gain = I_gain # dictionary of initial bandit parameters
    self.loss = I_loss
    self.po = I_payoff # total earned for that machine


class Click:
  def __init__(self,xcoord,ycoord):
    self.x = xcoord # x for every click on canvas
    self.y = ycoord # y for every click on canvas
```

```
class Control:

   def __init__(self,I_routeUse):

    self.route = I_routeUse

    self.playlimit= 250


class DecideTime:

  def __init__(self):

    self.start = time.time() # time since last decision


class Application(Frame):

  def __init__(self, master=None):

    Frame.__init__(self, master)

    self.pack()

    self.buildFrame()

    self.remaining = 0


  def restart(self, remaining = None):

    if remaining is not None:

      self.remaining = remaining

    if self.remaining <= 0:

      self.hurry.configure(text=“time’s up!”)

    else:

      self.hurry.configure(text=“%d” % self.remaining)

      self.remaining = self.remaining - 1
```

```
            self.after(1000, self.countdown)


          def buildFrame(self, remaining = None):
           # self.buildFrame2()
           self.customFont1 = tkFont.Font(family=“Arial Bold,” size=30)
           self.customFont2 = tkFont.Font(family=“Arial Bold,” size=20)
           self.customFont3 = tkFont.Font(family=“Arial Bold,” size=10)
           self.customFont4 = tkFont.Font(family=“Arial Bold,” size=20)


           topLabel = Label(self,text=“Select route for next convoy.,”
font=self.customFont2).grid(row=0,column=2, pady=25)
           #topLabel = Label(self,text=“Select route for next convoy.,”
font=self.customFont2).grid(row=0,column=1, columnspan=3, pady=25)
           #Label(self,text= ““).grid(row=5,column=2)
           Label(self,text= “Damage to Enemy Forces,”
font=self.customFont2).grid(row=30,column=1,pady=25)


           #Label(self,text=0, fg=“black,”
font=self.customFont1).grid(row=7,column=2)
           l1 = Label(self,textvariable=v_gain, fg=“black,”
font=self.customFont1).grid(row=25,column=1, pady = 0)
           Label(self,text= “Damage to Friendly Forces,”fg=“red,”
font=self.customFont2).grid(row=30,column=4)
           l2 = Label(self,textvariable=v_loss, fg=“red,”
font=self.customFont1).grid(row=25,column=4, pady = 0)
```

```
b_1 = Button(self,command=bdt1,bg=‘white’)
b_2 = Button(self,command=bdt2,bg=‘white’)
b_3 = Button(self,command=bdt3,bg=‘white’)
b_4 = Button(self,command=bdt4,bg=‘white’)

self.photo1=ImageTk.PhotoImage(file=“Picture2.png”)
self.photo2=ImageTk.PhotoImage(file=“Picture2.png”)
self.photo3=ImageTk.PhotoImage(file=“Picture2.png”)
self.photo4=ImageTk.PhotoImage(file=“Picture2.png”)

b_1.config(image=self.photo1, width=“400,”height=“400”)
b_2.config(image=self.photo2, width=“400,”height=“400”)
b_3.config(image=self.photo3, width=“400,”height=“400”)
b_4.config(image=self.photo4, width=“400,”height=“400”)

b_1.grid(row=6, column=1)
b_2.grid(row=6, column=2)
b_3.grid(row=6, column=3)
b_4.grid(row=6, column=4)

l30 = Label(self, text=,”“ fg=“gray50,”
font=self.customFont3,anchor=E).grid(row=3,column=1, pady=20)
```

```
          l35 = Label(self, text=,”“ fg=“gray50,”
font=self.customFont3,anchor=E).grid(row=7,column=2, pady=20)
          l36 = Label(self, text=,”“ fg=“gray50,”
font=self.customFont3,anchor=E).grid(row=8,column=2, pady=20)
          l3 = Label(self,text=“Accumulated Damage :,”fg=“gray50,”
font=self.customFont2,anchor=E).grid(row=1,column=2, pady=20)
          l4 = Label(self,textvariable=v_onhand, fg=‘gray50’,
font=self.customFont1).grid(row=1,column=3, pady=25)
          #l5 = Label(self,text=“(Positive number is good),”fg=“gray50,”
font=self.customFont2,anchor=E).grid(row=1,column=4)


        def callback(e):
         click.x = e.x
         click.y = e.y
         # print “clicked at,” e.x, e.y


        def WriteToFile(listArray, subName):
         with open(subName, ‘wb’) as csvfile:
          w = csv.writer(csvfile)
          w.writerow([‘trial’] + [‘routeSel’] + [‘trialGain’] + [‘trialLoss’] +[‘Damage’] +
[‘x’] + [‘y’] + [‘latent’] + [‘unixTime’]+ [‘machTime’] + [‘cogState’] +[‘CAPTTIM’])
          for e in listArray:   # for every trial data array,
           w.writerow(e)    # write it to file


        def readFromFile():
```

```
 with open(‘TDC.csv’, ‘rb’) as f:
  reader = csv.reader(f)
  for row in reader:
   latentlistR = row[7]
   choiceRead = row[1]

   print “route: ,” choiceRead, “latency: ,” latentlistR
   #print row[3]
#-------------------------------------------#
#      GLOBAL CONSTANTS       #
#-------------------------------------------#

runData = [] # storage tuple to temp store data
latentList = [] #store latency times
latentListR = []
latentList50 = [] #have to use something different for first 50 becuase we need the whole latentList later
avglatentList = [] #store EWMA latency times
avgLatencyTime = 0
latencyLambda = 0.9 #EWMA lambda parameter
sdLatencyTime = 0.0 #standard deviation of latency time
cogState = ‘test’ #Cognitive State string
baseLineLatency = 0
#List of intervention messages
```

```
messageList = [“Score could be better; attend to friendly damage,”
    “Score could be better; attend to friendly damage and try other routes,”
    “Score is looking good; go ahead and make decisions quickly,”
    “Score is looking good; stay with your strategy”]
CAPTTIM = ‘ ‘
gainList = [] #Capture absolute regret
medGain = [] #Capture median regret values
subName = strftime(“%Y %b %d %a %H %M Mil MultiArmBandit.csv,” localtime()) # time as file name
root = Tkinter.Tk( )
# Player parameters
x = Player(2000,0) # instantiate player object onhand,plays
irouteUse= {}
irouteUse[1]= 0
irouteUse[2]= 0
irouteUse[3]= 0
irouteUse[4]= 0
game = Control(irouteUse)
time_to_decide = DecideTime()

click = Click(0,0)
v_onhand = DoubleVar() # instantiate running total onhand
v_onhand.set(x.oh) # running total from all machines
```

```
Dcolor = ‘black’

v_gain = DoubleVar()

v_gain.set(0) # running total from all machines

v_loss = DoubleVar()

v_loss.set(0) # running total from all machines

v_plays = IntVar()

#v_plays.set(game.playlimit-x.p)

v_plays.set(x.p)


v_bdt1 = DoubleVar() # last payoff value for machine 1

v_bdt2 = DoubleVar()

v_bdt3 = DoubleVar()

v_bdt4 = DoubleVar()


v_gain1 = DoubleVar()

v_gain2 = DoubleVar()

v_gain3 = DoubleVar()

v_gain4 = DoubleVar()


# Bandit parameters held in a dictionary

igain= {}

igain[1]= 100 #bandit 1: n1,p1,n2,p2

igain[2]= 100

igain[3]= 50
```

```
        igain[4]= 50


        iloss= {}

        iloss[1]=    [-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-
150,

              0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,

              -150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,

              -300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,

              -200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,

              -250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,

              -350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,

              0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-
150,0,0

              -350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,

              0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,

              -150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,

              -300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,

              -200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,

              -250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-150,0,0,

              -350,-250,0,-200,0,-300,0,-150,

              0,0,-350,-250,0,-200,0,-300,0,-150,0,0,-350,-250,0,-200,0,-300,0,-
150,0,0

              -350,-250,0,-200,0,-300,0,-150,0,0,-350,-250]
```

```
    iloss[2]=            [-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-
1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,

        -1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0,-1250,0,0,0,0,0,0,0,0,0]


    iloss[3]=   [-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-
50,0,-50,0,

        -50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-
50,0,0,

        -50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-
50,

        0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-
50,0,

        0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,

        -50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,

        -50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,

        -50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,

        -50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,
        0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,
        -50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,
        -50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,-50,0,-50,0,-50,0,-50,0,0,-50,
        -50,-50,0,-50,0,-50,0,-50,0]


    iloss[4]=                    [-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-
250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,
        -250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0,-250,0,0,0,0,0,0,0,0,0]
    print ‘Deck A is’,len(iloss[1]),’cards long.’
    print ‘Deck B is’,len(iloss[2]),’cards long.’
    print ‘Deck C is’,len(iloss[3]),’cards long.’
    print ‘Deck D is’,len(iloss[4]),’cards long.’


    ipayoff= {}
    ipayoff[1]= 0
    ipayoff[2]= 0
```

```
ipayoff[3]= 0

ipayoff[4]= 0


b=Bandit(igain,iloss,ipayoff)


v_bdt1.set(b.po[1])

v_bdt2.set(b.po[2])

v_bdt3.set(b.po[3])

v_bdt4.set(b.po[4])


v_gain1.set(0)

v_gain2.set(0)

v_gain3.set(0)

v_gain4.set(0)


def refresh():

  app.mainloop()


def bdt1():

  machine= 1

  gain=b.gain[machine]

  loss=b.loss[machine]

  getGain(gain,loss,machine)
```

```
def bdt2():
  machine= 2
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)

def bdt3():
  machine= 3
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)

def bdt4():
  machine= 4
  gain=b.gain[machine]
  loss=b.loss[machine]
  getGain(gain,loss,machine)

#Display a modal pop-up info box with the supplied message string
def displayDialog(message):
  tkMessageBox.showinfo("Guidance," message)

def getGain(gain,loss,mach):
  CAPTTIM = ''
```

```
cogState = ‘‘
gainP = gain
lossP = -1*loss.pop()
gain = gainP - lossP
latent = time.time() - time_to_decide.start
time_to_decide.start = time.time()
game.route[mach] += 1
b.po[mach] = b.po[mach] + gainP + lossP # update earning by machine
x.oh = x.oh + gain  # update earnings total, subtracting any cost to play
if x.oh < 0:
  Dcolor = ‘red’
else:
  Dcolor = ‘black’
v_onhand.set(x.oh)
x.p = x.p + 1         # update times game played
dt = datetime.now()
machTime= dt.time()
unixTime = calendar.timegm(dt.utctimetuple())

if x.p <= 50:
  selData=[x.p, mach, gainP,lossP,x.oh, click.x, click.y,
latent,unixTime,machTime, CAPTTIM]
  runData.append(selData)     # store data
```

```
################################
#   Cognitive State     #
################################


 if x.p<=2:
  avgLatencyTime = latent
  avglatentList.append(latent) #Store Average Latency Time
  latentList.append(latent)
  latentList50.append(latent)
 else:
  if gain >= 0: ##only append the latency time to the list if the choice is not ‘bad’
   latentList.append(latent)
   latentList50.append(latent)
  #Compute EWMA Latency from Nesbitt Understanding Optimal Decision Making
   avgLatencyTime = latencyLambda*latentList[len(latentList)-1] + (1-latencyLambda)*avglatentList[len(avglatentList)-2]
  #Store Average Latency Time of these GOOD decisions
   avglatentList.append(avgLatencyTime)
  else:
   avgLatencyTime = latencyLambda*latentList[len(latentList)-1] + (1-latencyLambda)*avglatentList[len(avglatentList)-2]
   #Still computing the average latency time, just not appending it to the list when subject takes a hit during first 50 trials
```

```
      print “bad choice, latency not added to avgLatentList”


    baseLineLatency = np.mean(latentList50)


   else:
    latentList.append(latent) #still have to capture all the raw times? We should be using EWMA for Exp/Exp
    baseLineLatency = np.mean(latentList50) ##there’s nothing added to latentList after trial 50, so baseLineLatency stays the same
    STDofBaseLineLatency = np.std(latentList50) #Compute the standard deviation of the latency time
    avgLatencyTime = latencyLambda*latentList[len(latentList)-1] + (1-latencyLambda)*avglatentList[len(avglatentList)-2]
    avglatentList.append(avgLatencyTime)
    print “SD of baseline,” STDofBaseLineLatency
    #get the mean of the last 10 latency values from the overall list
    Last10AvgLatencies = avglatentList[len(avglatentList)-10:len(avglatentList)]
    STDLast10 = np.std(Last10AvgLatencies)
    print “STD of Last 10 trials: ,” STDLast10


    if STDLast10 <= 1.5*STDofBaseLineLatency:
      cogState = ‘Exploit’
    elif STDLast10 > 1.5*STDofBaseLineLatency:
      cogState = ‘Explore’
```

```
  print “CogState is: %s” %cogState


##################################
#    Regret       #
##################################


gainList.append(gain)
regret = -gain
for check in range(50,game.playlimit,10):
  checkLast10 = gainList[len(gainList)-10:len(gainList)] #take the most recent 10 gain values from the overall list
  checkLast15 = gainList[len(gainList)-15:len(gainList)] #take the most recent 15 gain values from the overall list
  averageLast10 = np.average(checkLast10)
  medianLast10 = np.median(checkLast10) #the median of the above list to compare against most recent trial
  medianLast15 = np.median(checkLast15)
  averageLast15 = np.average(checkLast15)


  if x.p == check:
    print “The last 10 gains: ,” checkLast10
    print “median of last 10 trials: ,” medianLast10
    print “average of last 10 trials ,” averageLast10
```

```
      if averageLast10 < medianLast10 and cogState == ‘Explore’:
        CAPTTIM = “YELLOW” #the inequality above “ave > median” is the definintion of gain (note line 381 that regret is opposite of gain)
        displayDialog(messageList[0])
      elif averageLast10 < medianLast10 and cogState == ‘Exploit’:
        CAPTTIM = “RED”
        displayDialog(messageList[1])
      elif averageLast10 >= medianLast10 and cogState == ‘Explore’:
        CAPTTIM = “ORANGE”
        displayDialog(messageList[2])
      elif averageLast10 >= medianLast10 and cogState == ‘Exploit’:
        CAPTTIM = “GREEN”
        displayDialog(messageList[3])
      print “CAPTTIM,” CAPTTIM

## Compute CAPTTIM for every trial and append it to the selection data for later evaluation of proportion of time in R/Y/O/G
  if x.p>50:
    if averageLast10 < medianLast10 and cogState == ‘Explore’:
      CAPTTIM = “YELLOW”
    elif averageLast10 < medianLast10 and cogState == ‘Exploit’:
      CAPTTIM = “RED”
    elif averageLast10 >= medianLast10 and cogState == ‘Explore’:
      CAPTTIM = “ORANGE”
```

```
 elif averageLast10 >= medianLast10 and cogState == ‘Exploit’:
  CAPTTIM = “GREEN”
 selData=[x.p, mach, gainP,lossP,x.oh, click.x, click.y, latent,unixTime,machTime,cogState,CAPTTIM]
 runData.append(selData)


v_gain.set(0)
v_loss.set(0)
if mach == 1:
 v_bdt1.set(b.po[mach])
 v_gain1.set(gain)
if mach == 2:
 v_bdt2.set(b.po[mach])
 v_gain2.set(gain)
if mach == 3:
 v_bdt3.set(b.po[mach])
 v_gain3.set(gain)
if mach == 4:
 v_bdt4.set(b.po[mach])
 v_gain4.set(gain)


v_plays.set(x.p)
v_gain.set(gainP)
v_loss.set(-1*lossP)
```

```
  if x.p >= game.playlimit:
    print (“\n\nPLAY LIMIT MET\n\n”)
    WriteToFile(runData,subName)
    print (‘shut down’)
    root.quit()\


if __name__ == “__main__”:
  app = Application(master=root)
  app.master.title(“Route Selection and Battle Damage Tool”)
  app.master.minsize(1000,400)
  root.bind(“<1>,” callback)
  app.mainloop()
  root.destroy()
```

APPENDIX C. DEMOGRAPHIC SURVEY

Convoy Task

Demographic Survey

Subject number: Date:

1. Age: _________
2. Gender: Female_________ Male_________
3. Preferred hand for writing: Left:_________ Right:_________
4. Are you currently serving in the Armed Forces: Yes No
    a. Which branch:_________________
    b. Years of service: _________
    c. Highest rank:_________
    d. Have you deployed to a combat zone (receipt of Imminent Danger Pay)?

        No (skip to e.) Yes (i – iii below)

        i. Date of return from latest deployment____________
        ii. Role during deployment (e.g. Surface Warfare Officer, Engineer Company Commander, Division Logistics Officer, AH-1W section lead, etc.) ____
        iii. Responsibilities (Route clearance, Fires planner, etc):

            ____

    e. If no combat deployment, what was your billet/rate immediately prior to NPS?

        ____

    f. If no combat deployment, what were your responsibilities immediately prior to NPS? ____

        ____

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. POST TASK SURVEY

Convoy Task

Post Task Survey

Subject number: Date:

1. During the Convoy Task how did you determine which route to select?

2. If you used a particular strategy what was it?

    a. Did your strategy change during the task?

    b. If yes, at about which point (e.g. right away, about halfway, toward the end) did you change your approach?

    c. If yes, what caused you to change your approach?

3. Rank the routes overall from safest (1) to most dangerous (4):

Left	Center Left	Center Right	Right

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Archer, J. C. (2010). State of the science in health professional education: effective feedback. *Medical Education*: *44*: 101–108.

Bechara, A., Damasio, A. R., Damasio, H., & Anderson, S. W. (1994). Insensitivity to future consequences following damage to human prefrontal cortex. *Cognition*, *50*(1), 7–15.

Bechara, A., Damasio, H., Tranel, D. & Damasio, A. R. (1997). Deciding Advantageously before knowing the advantageous strategy. *Science, 275*, 1293–1294.

Canas, J. J., Quesada, J. F., Antoli, A. & Fajardo, I. (2003).Cognitive flexibility and adaptability to environmental changes in dynamic complex problem solving tasks. *Ergonomics, 45*(5), 482–501.

Chickering, A. W., & Gamson, Z. F. (1987). Seven principles for good practice in undergraduate education. *American Association for Higher Education Bulletin*, 3–7.

Crichton, M., Flin, R. (2001). Training for emergency management: tactical decision games. *Journal of Hazardous Materials 88*, 255–266.

Critz, J. W. (2015). Understanding optimal decision making. (Master's thesis). Retrieved from http://calhoun.nps.edu/handle/10945/45832

Kennedy, Q., Adamson, M., Huston, J. & Nesbitt, P. (2015). Can a Simple Wargame Provide an Unobtrusive Indicator of TBI? A Case Study. Presented at 43rd Annual Meeting of the International Neuropsychological Society, Denver, CO.

Kennedy, Q., Nesbitt, P., Alt, J., & Fricker, R. D. (2015). Cognitive Alignment with Performance Targeted Training Intervention Model: CAPTTIM. *Human Factors and Ergonomics Society 2015 Annual Meeting*. 59(1), 95 – 99.

Krain, A. L., Wilson, A. M., Arbuckle, R., Castellanos, F. X., & Milham, M. P. (2006). Distinct neural mechanisms of risk and ambiguity: a meta-analysis of decision-making. *NeuroImage*, *32*(1), 477–484. doi:http://dx.doi.org/10.1016/j.neuroimage.2006.02.047

Lyster, R. & Ranta, L. (1997). Corrective feedback and learner uptake. *Studies in Second Language Acquisition*. 37–66.

Nesbitt, P., Kennedy, Q., Alt, J, & Fricker, R. (2015). Iowa Gambling Task Modified for Military Domain. *Military Psychology*, *27*(4), 252260.

Nesbitt, P., Kennedy, Q., Alt, J., Yang, J., Fricker, R., Appleget, J., Huston, J., Patton, S., Whitaker, L. (2013). Understanding Optimal Decision-making in Wargaming. Monterey, CA: Naval Postgraduate School. NPS-OR-14-001.

Odierno, R., & McHugh, J. (2015). Army human dimension strategy Draft Version 5.3. U.S. Army Publication. Retrieved from http://usacac.army.mil/sites/default/files/publications/20150524_Human_Dimension_Strategy_vr_Signature_WM_1.pdf

Sacchi, S. (2014). Iowa Gambling Task app. Retrieved from https://play.google.com/store/apps/details?id=com.zsimolabs.iowa&hl=en

Steingroever, H. and Wetzels, R., Horstmann, A. Neumann, J., & Wagenmakers, E-J. (2013). Performance of healthy participants on the Iowa Gambling Task. *Psychological Assessment*, *25*(1): 180 – 193.

Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction.* Cambridge, MA: MIT Press.

Taylor, W (2000). Change-Point Analyzer 2.0 shareware program, Taylor Enterprises, Libertyville, Illinois. Web: http://www.variation.com/cpa.

U.S. Marine Corps. (2012). 2012 U.S. Marine Corps S&T Strategic Plan. Arlington, VA.

Wickens, C. D. & Alexander, A.L. (2009). Attentional Tunneling and Task Management in Synthetic Vision Displays. *The International Journal of Aviation Psychology,* 19(2), 182 – 199.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California